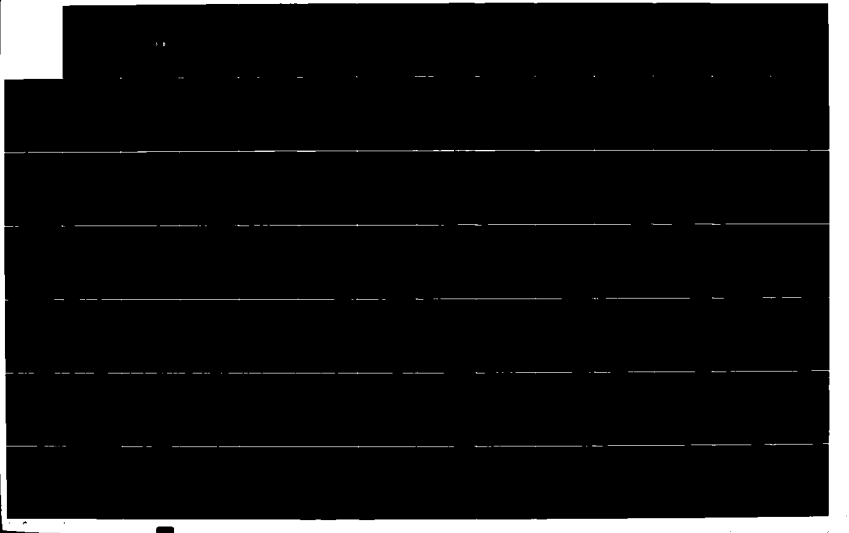
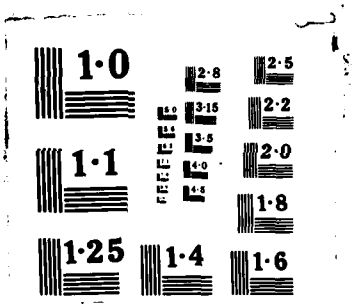
AD-A167 396 ADA (TRADE NAME) COMPILER VALIDATION SUMMARY REPORT 1/1
ALSYS ALSYCOMP 001 VE..(U) BUREAU D' ORIENTATION DE LA
NORMALISATION EN INFORMATIQUE ROCQ.. 08 NOV 85
UNCLASSIFIED ADA-85.4 F/G 9/2 NL




1·0
1·1
1·25
1·4
1·6
1·8
2·0
2·2
2·5
2·8
3·15
3·5
4·0

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1 REPORT NUMBER	2 GOVT ACCESSION NO. / 3 RECIPIENT'S CATALOG NUMBER
4 TITLE (and Subtitle) Ada Compiler Validation Summary Report: AlsyCOMP_001, Version 1.3, VAX-11/750 host, Altos ACS 68000 14 target.	5 TYPE OF REPORT & PERIOD COVERED 8 November '85 to 8 Nov. 1986
	6 PERFORMING ORG. REPORT NUMBER Ada 85.4
7 AUTHOR(s) BNI/AVF	8 CONTRACT OR GRANT NUMBER(s)
9 PERFORMING ORGANIZATION NAME AND ADDRESS BNI/AVF Domaine de Voluceau - Rocquencourt B.P. 105 - 78153 LE CHESNAY CEDEX, FRANCE	10 PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11 CONTROLLING OFFICE NAME AND ADDRESS Ada Joint Program Office 1211 S. Fern Street, Rm. C-107 Arlington, VA 22202	12 REPORT DATE 8 November 1985
	13 NUMBER OF PAGES 95
14 MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) BNI/AVF	15 SECURITY CLASS. (of this report) UNCLASSIFIED
	15a DECLASSIFICATION/DOWNGRADING SCHEDULE

16 DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17 DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Unclassified

DTIC ELECTE MAY 01 1986 S D

18 SUPPLEMENTARY NOTES

19 KEY WORDS (Continue on reverse side if necessary and identify by block number)

Ada Programming language, Ada Compiler Validation Summary Report, Ada Compiler Validation Capability, ACVC, Validation Testing, Ada Validation Office, AVO, Ada Validation Facility, AVF, ANSI/MIL-STD-1815A, Ada Joint Program Office, AJPO.

20 ABSTRACT (Continue on reverse side if necessary and identify by block number)

See attached.

DTIC FILE COPY

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD-A167 396

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

ABSTRACT

This Validation Summary Report presents the results and conclusions of testing performed on the AlsyCOMP_005, version 1.0. Standardized tests serve as input to an Ada compiler, producing results which are evaluated by the validation team. This summary briefly states the highlights of the AlsyCOMP_005, version 1.0 validation.

On-site testing was performed 31 October 1985 through 2 November 1985 at Alsys premises in La Celle Saint Cloud - France, under the auspices of the BNI (AVF), according to Ada Validation Office policies and procedures. The AlsyCOMP_005, version 1.0 is hosted on SUN Workstation 2/120 and also on a SUN Workstation 2/50 operating under SUN UNIX 4.2 release 2.0, it is also hosted on SUN workstation 3/160 operating under SUN UNIX 4.2 release 3.0. The suite of tests known as the Ada Compiler Validation Capability (ACVC), Version 1.6, was used. The ACVC is used to validate conformance of a compiler to ANSI/MIL-STD-1815A Ada. The purpose of testing is to ensure that a compiler properly implements legal language constructs and that it identifies and rejects illegal language constructs. The testing also identifies behavior that is implementation dependent but permitted by the Ada Standard. Six classes of tests are used. These tests are designed to perform checks at compile time, at link time, or during execution.

The results of validation are summarized in the following table.

RESULT	TEST CLASS						TOTAL
	A	B	C	D	E	L	
Passed	60	777	961	16	8	1	1823
Failed	0	0	0	0	0	0	0
Inapplicable	1	5	267	1	0	2	276
Anomalous	0	0	0	0	0	0	0
Withdrawn	0	18	45	0	0	0	63
TOTAL	61	800	1273	17	8	3	2162

Ada is a registered trademark of the United States Government (Ada Joint Program Office)

86 4 29 121

Ada COMPILER VALIDATION SUMMARY REPORT:

ALSYS
AlsyCOMP_001, version 1.3
VAX-11/750 host,
Altos ACS 68000 14 target

Completion of On-Site Validation:
8 November 1985

Prepared By:
BNI/AVF
Domaine de Voluceau - Rocquencourt
B.P.105 - 78153 LE CHESNAY CEDEX
FRANCE

Prepared For:
Ada Joint Program Office
United States Department of Defense
Washington, D.C.

Ada is a registered trademark of the United States Government
(Ada Joint Program Office)

+++++++++++++++++++++++++
+ +
+ Place NTIS form here +
+ +
+++++++++++++++++++++++++

Ada Compiler Validation Summary Report:

Compiler Name: AlsyCOMP_001, version 1.3

Host Computer	Target Computer
VAX-11/750	ALTOS ACS 68000 14
under	under
VMS - version 4.1	ALTOS Operating system version 1

Testing Completed 8 November 1985 Using ACVC 1.6

This report has been reviewed and approved:

Ada Validation Facility
BNI
Nicolas Malagardis represented by Jacqueline Sidi
Domaine de Voluceau - Rocquencourt
B.P.105 - 78153 LE CHESNAY CEDEX
FRANCE

Acting as the
Ada Validation Office (AVO)
John F. Kramer, Jr.
Institute for Defense Analyses
Alexandria, VA

Ada Joint Program Office (AJPO)
Virginia L. Castor
Director
Washington, D.C.

Accesion For	
NTIS CRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and / or Special
A-1	23

Ada is a registered trademark of the United States Government
(Ada Joint Program Office)

EXECUTIVE SUMMARY

This Validation Summary Report presents the results and conclusions of testing performed on the AlsyCOMP_001, version 1.3. Standardized tests serve as input to an Ada compiler, producing results which are evaluated by the validation team. This summary briefly states the highlights of the AlsyCOMP_001, version 1.3 validation.

On-site testing was performed 31 October 1985 through 8 November 1985 at Alsys premises in La Celle Saint Cloud - France, under the auspices of the BNI (AVF), according to Ada Validation Office policies and procedures. No precise timing information could be collected as numerous problems arised in transfering the files from one machine to the other. This had for consequence several days delay. The AlsyCOMP_001, version 1.3 is hosted on VAX-11/750 operating under VMS version 4.1. The suite of tests known as the Ada Compiler Validation Capability (ACVC), Version 1.6, was used. The ACVC is used to validate conformance of a compiler to ANSI/MIL-STD-1815A Ada. The purpose of testing is to ensure that a compiler properly implements legal language constructs and that it identifies and rejects illegal language constructs. The testing also identifies behavior that is implementation dependent but permitted by the Ada Standard. Six classes of tests are used. These tests are designed to perform checks at compile time, at link time, or during execution.

The results of validation are summarized in the following table.

RESULT	TEST CLASS						TOTAL
	A	B	C	D	E	L	
Passed	60	777	961	17	8	1	1824
Failed	0	0	0	0	0	0	0
Inapplicable	1	5	267	0	0	2	275
Anomalous	0	0	0	0	0	0	0
Withdrawn	0	18	45	0	0	0	63
TOTAL	61	800	1273	17	8	3	2162

Ada is a registered trademark of the United States Government (Ada Joint Program Office)

Tests found to contain errors were withdrawn from Version 1.6 of the Ada Compiler Validation Capability (ACVC). When validation was completed, the following tests had been withdrawn:

B38105B-AB	C45521A..Y-B (25 tests)	C48005C-B
C48006B-B	C64103C-B	C64103D-B
C64105E-AB	C64105F-AB	B66001A-B
B67001A-B	B67004A-B	B74103F-B
B74207A-B	C93005B-B	C93005C-B
C93007B-B	BC3220B-B	CA2009E-B
CA1003B-AB	CA1011A*-B	CA1108A-B
CA1108B-B	CA2009B-B	CA2009F*-B
BC1013A-B	BC3204A..D-B (4 tests)	BC3205A..D*-B (4 tests)
BC3405B-B	BC3503A-B	CE2107E-B
CE3603A-B	CE3604A-B	CE3704M-B

Some tests demonstrate that language features are not supported by an implementation. For this implementation the tests determined the following.

. SHORT_FLOAT is not supported:

B86001CP-AB.DEP C34001F-B.DEP C35702A-AB.DEP

. LONG_FLOAT is not supported:

B86001CQ-AB.DEP C34001G-B.DEP C35702B-AB.DEP

. Representation specifications for noncontiguous enumeration representations are not allowed:

C55B16A-AB.DEP

. No other integer type other than INTEGER, SHORT_INTEGER, AND LONG_INTEGER is supported:

B86001DT-AB.DEP

. The package SYSTEM is used by package TEXT_IO:

C86001F-B.ADA

. The 'SIZE clause is not supported:

C87B62A-B.DEP

. The 'STORAGE_SIZE clause is not supported:

C87B62B-B.DEP

. The 'SMALL clause is not supported:

C87B62C-B.DEP

. Generic package bodies cannot be compiled in separate compilation files:

CA2009C*-B.DEP

. Pragma INLINE is not supported for procedures:

LA3004A*-AB.ADA

. Pragma INLINE is not supported for functions:

LA3004B*-B.DEP

ACVC Version 1.6 was taken on-site via magnetic tape to Alsys premises in La Celle Saint Cloud - France. The tape was loaded, and all tests, except the withdrawn tests and any executable tests which make use of a floating point precision greater than SYSTEM.MAX_DIGITS, were compiled on VAX-11/750. Class A, C, D, and E tests were executed on the ALTOS.

On completion of testing, all results were analyzed for failed Class A, C, D, or E programs, and all Class B and L compilation results were individually analyzed.

The ACVC, Version 1.6, contains 2162 tests of which 1824 were applicable to AlsyCOMP_001, version 1.3. 21 tests were processed although inapplicable. No anomalies were found in the testing of this compiler. Testing demonstrated that all applicable tests were passed by this compiler. The AVF concluded that the results show acceptable compliance to ANSI/MIL-STD-1815A Ada.

CHAPTER 1

INTRODUCTION

The Validation Summary Report describes how an Ada compiler conforms to the language standard. This report explains all technical terms used within and thoroughly reports the Ada Compiler Validation Capability (ACVC) test results. Ada compilers must be written according to the *language specification* as given in the ANSI/MIL-STD-1815A Ada. All implementation-defined features must be included for the compiler to conform to the Standard. Following the guidelines of the Standard ensures continuity between compilers. That is, the entire Standard must be implemented, and nothing can be implemented that is not in the Standard.

Even though all validated Ada compilers conform to the Standard, it must be *understood* that some differences do exist between implementations. ANSI/MIL-STD-1815A permits some implementation dependencies, e.g., the maximum length of identifiers, the maximum values of integer types, etc. These implementation-dependent features limit the portability of programs between compilers. Other differences between compilers are due to limitations imposed on a compiler by the operating system and by the hardware. All of these dependencies are given in the report.

Validation summary reports are written according to a standardized format. Compiler users can, therefore, more easily compare the reports from several compilers when selecting a compiler for a given task. The validation report can be completed mostly from the test results produced during validation testing. Additional testing information is given at the end of the report and states problems and details which are unique for a specific compiler. The format of the validation report limits variance between reports, enhances readability of the report, and accelerates report readiness.

1.1- Purpose of this Validation Summary Report

The Validation Summary Report documents the results of the testing performed on an Ada compiler. Testing was carried out for the following *purposes*:

- To identify any language constructs supported by the translator that do not conform to the Ada Standard
- To identify any unsupported language constructs required by the Ada Standard

. To describe the implementation-dependent behavior allowed by the Ada Standard

Testing of this compiler was conducted by BNI according to policies and procedures established by the Ada Validation Office (AVO). Testing was conducted from 31 October 1985 through 8 November 1985 at Alsys premises in La Celle Saint Cloud - France. No precise timing information could be collected as numerous problems arised in transfering the files from one machine to the other. This had for consequence several days delay.

1.2- Use of this Validation Summary Report

Consistent with the national laws of the originating country, the Ada Validation Office may make full and free public disclosure of this report. In the United States, this is provided in accordance with the "Freedom of Information Act" (5 U.S.C. #552). The results of this validation apply only to the computers, operating systems, and compiler versions identified in this report.

The organizations represented on the signature page of this report do not represent or warrant that any statement or statements set forth in this report are accurate or complete, or that the subject compiler has no nonconformances to the Ada Standard other than those presented. This report is not intended for the purpose of publicizing the findings summarized herein.

Questions regarding this report or the validation tests should be directed to:

Ada Validation Office
Institute for Defense Analyses
1801 N. Beauregard
Alexandria VA 22311

and to:

BNI
Domaine de Voluceau - Rocquencourt
B.P.105 - 78153 LE CHESNAY CEDEX
FRANCE

1.3- References

. Reference Manual for the Ada Programming Language, ANSI/MIL-STD-1815A, Feb 1983.

. Ada Validation Organization Policies and Procedures
T.H. Probert, MITRE Corporation, MTR-82W00103, June 1982

. Ada Compiler Validation Capability Implementers' Guide,
SofTech, Inc., Dec 1984.

1.4- Definition of Terms

Anomaly	A test result that, given pre-validation analysis, is not expected during formal validation but is judged allowable under the circumstances.
ACVC	The Ada Compiler Validation Capability. A set of programs that evaluates the conformance of a compiler to the Ada language specification, ANSI/MIL-STD-1815A.
Ada Standard	ANSI/MIL-STD-1815A, February 1983.
Applicant	The agency requesting validation.
AVF	The BNI. In the context of this report, the AVF is responsible for conducting compiler validations according to established policies and procedures.
AVO	The Ada Validation Office. In the context of this report, the AVO is responsible for setting policies and procedures for compiler validations.
Compiler	A processor for the Ada language. In the context of this report, a compiler is any language processor, including cross-compilers, translators, and interpreters.
Failed test	A test for which the compiler generates a result that demonstrates nonconformance to the Ada Standard.
Host	The computer on which the compiler resides.
Inapplicable test	A test that uses features of the language that a compiler is not required to support or may legitimately support in a way other than the one expected by the test.
Passed test	A test for which a compiler generates the expected result.
Target	The computer for which a compiler generates code.
Test	A program that evaluates the conformance of a compiler to a language specification. In the context of this report, the term is used to designate a single ACVC test. The text of a program may be the text of one or more compilations.

Withdrawn test	A test that has an invalid test objective, fails to meet its test objective, or contains illegal use of the language.

1.5- Configuration

The candidate compilation system for this validation was tested under the configuration:

Compiler: AlsyCOMP_001, version 1.3

Test Suite: Ada Compiler Validation Capability, Version 1.6

Host Computer:

Machine(s):	VAX-11/750
Operating System:	VMS - version 4.1
Memory Size:	6 Megabytes
Disk System:	456 Megabytes

Target Computer:

Machine(s):	ALTOS ACS 68000 14
Operating System:	ALTOS Operating system version 1
Memory Size:	1 Megabyte
Disk System:	40 Megabytes

CHAPTER 2

TEST RESULTS

2.1- ACVC Test Classes

Conformance to ANSI/MIL-STD-1815A is measured using the Ada Compiler Validation Capability (ACVC). The ACVC contains both legal and illegal Ada programs structured into six test classes: A, B, C, D, E, and L. Legal programs are compiled and executed while illegal programs are just compiled. Support packages are used to report the results of the legal programs. A compiler must correctly process each of the tests in the suite and demonstrate conformance to the Ada Standard by either meeting the pass criteria given for the test or by showing that the test is inapplicable to the implementation. Tests that are found to contain errors are withdrawn from the ACVC. Detailed test results are listed in the Appendix D. The results of validation testing are summarized in the following table:

RESULT	TEST CLASS						TOTAL
	A	B	C	D	E	L	
Passed	60	777	961	17	8	1	1824
Failed	0	0	0	0	0	0	0
Inapplicable	1	5	267	0	0	2	275
Anomalous	0	0	0	0	0	0	0
Withdrawn	0	18	45	0	0	0	63
TOTAL	61	800	1273	17	8	3	2162

A total of 1845 tests were processed during this validation attempt. The 63 withdrawn tests in Version 1.6 were not processed, nor were 254 Class C tests that were inapplicable because they use floating point types having digits that exceed the maximum value for the implementation. All other tests were processed.

Some conventions are followed in the ACVC to ensure that the tests are reasonably portable without modification. For example, the tests make use of only the basic 55 character set, contain lines with a maximum length of 72 characters, use small numeric values, and place features that may not be supported in separate tests. However, some tests contain values that require the test to be customized according to implementation-specific values. The values used for this validation are listed in Appendix B.

2.1.1- Class A Tests

Class A tests check that legal Ada programs can be successfully compiled and executed. However, no checks are performed during execution to see if the test objective has been met. For example, a Class A test checks that reserved words of another language other than those already reserved in the Ada language) are not treated as reserved words by an Ada compiler. A Class A test is passed if no errors are detected at compile time and the program executes to produce a message indicating that it has passed. If a Class A test cannot be compiled and executed because of its size, then the test is split into a set of smaller subtests that can be processed. A split was required for 1 test:

AE2101A-B.ADA

The following table shows that all applicable Class A tests were passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	13	6	0	5	2	12	13	2	0	0	0	7	60
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	0	0	0	0	0	0	0	1	0	0	0	0	1
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL	13	6	0	5	2	12	13	3	0	0	0	7	61

2.1.2- Class B Tests

Class B tests check that a compiler detects illegal language usage. Class B tests are not executable. Each test in this class is compiled and the resulting compilation listing is examined manually to verify that every syntax or semantic error in the test is detected. A Class B test is passed if every illegal construct that it contains is detected by the compiler. If one or more errors are not detected, then a version of the test is created that contains only the undetected errors. The resulting "split" is compiled and examined. The splitting process continues until all errors are detected by the compiler. Splits were required for 15 tests:

B32202A-B.ADA	B32202B-B.ADA	B32202C-B.ADA
B33006A-B.ADA	B37004A-B.ADA	B43201D-B.ADA
B45102A-AB.ADA	B61012A-B.ADA	B62001B-AB.ADA
B62001C-AB.ADA	B62001D-AB.ADA	B91004A-B.ADA
BA2001E0M-AB.ADA	BA2001E1-AB.ADA	BA2001E2-AB.ADA

The following table shows that all applicable Class B tests were passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	35	72	83	113	70	55	49	91	36	8	147	18	777
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	0	0	0	0	0	0	3	1	0	0	1	0	5
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	1	0	0	3	2	0	0	0	0	12	0	18
TOTAL	35	73	83	113	73	57	52	92	36	8	160	18	800

2.1.3- Class C Tests

Class C tests check that legal Ada programs can be correctly compiled and executed. Each Class C test is self-checking and produces a PASS/FAIL message indicating the result when it is executed. If a Class C test cannot be compiled because it exceeds the compiler's capacity, then the test is split into smaller subtests until all are compiled and executed. No splits were required.

The following table shows that all applicable Class C tests were passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	19	89	153	115	70	14	93	106	35	20	55	192	961
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	23	119	116	4	0	0	4	0	1	0	0	0	267
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	0	27	0	4	0	0	3	7	0	0	4	45
TOTAL	42	208	296	119	74	14	97	109	43	20	55	196	1273

2.1.4- Class D Tests

Class D tests check the compilation and execution capacities of a compiler. Since there are no requirements placed on a compiler by the Ada Standard for the number of identifiers permitted in a compilation, the number of units in a library, the number of nested loops in a subprogram body, and so on, a compiler may refuse to compile a Class D test. Each Class D test is self-checking and produces a PASS/FAIL message indicating the result when it is executed. If a Class D test fails to compile because the capacity of the compiler is exceeded, then the test is classified as inapplicable.

The following table shows that all applicable Class D tests were passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	1	0	4	9	3	0	0	0	0	0	0	0	17
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	0	0	0	0	0	0	0	0	0	0	0	0	0
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL	1	0	4	9	3	0	0	0	0	0	0	0	17

Capacities measured by the Class D tests are detailed in section 2.4, IMPLEMENTATION CHARACTERISTICS.

2.1.5 Class E Tests

Class E tests provide information about the compiler in those areas in which the Ada Standard permits implementations to differ. Each Class E test is executable and produces messages that indicate how the Ada Standard is interpreted. However, in some cases the Ada Standard permits a compiler to detect a condition either at compile time or at execution time, and thus a Class E test may correctly fail to execute. A Class E test is passed if it fails to compile and appropriate error messages are issued, or if it executes properly and produces a message that it has passed. If a Class E test cannot be compiled and executed because of its size, then the test is split into a set of smaller subtests that can be processed. No splits were required.

The following table shows that all applicable Class E tests were passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	1	3	2	1	0	0	0	0	0	0	0	1	8
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	0	0	0	0	0	0	0	0	0	0	0	0	0
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL	1	3	2	1	0	0	0	0	0	0	0	1	8

Information obtained from the Class E tests is detailed in section 2.4, IMPLEMENTATION CHARACTERISTICS.

2.1.6- Class L Tests

Class L tests check that incomplete or illegal Ada programs involving multiple, separately compiled units are detected and not allowed to execute. Class L tests are compiled separately and execution is attempted. A Class L test passes if it is rejected at link time and the test does not execute.

The following table shows that all applicable Class L tests were passed:

RESULT	CHAPTER												
	2	3	4	5	6	7	8	9	10	11	12	14	TOTAL
Passed	0	0	0	0	0	0	0	0	1	0	0	0	1
Failed	0	0	0	0	0	0	0	0	0	0	0	0	0
Inapplicable	0	0	0	0	0	0	0	0	2	0	0	0	2
Anomalous	0	0	0	0	0	0	0	0	0	0	0	0	0
Withdrawn	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL	0	0	0	0	0	0	0	0	3	0	0	0	3

2.1.7- Support Units

Three packages support the self-checking features of Class C tests: REPORT, CHECK_FILE, and VAR_STRINGS. The REPORT package provides the mechanism by which executable tests report results. It also provides a set of identity functions that are used to defeat some compiler optimization strategies to cause computations to be made by the target computer instead of the compiler on the host computer. The CHECK_FILE package is used to check the contents of text files written by some of the Class C tests for Chapter 14 of the Ada Standard. The VAR_STRINGS package defines types and subprograms for manipulating varying-length character strings. The operation of these three packages is checked by a set of executable tests. These tests produce messages that are examined manually to verify that the packages are operating correctly. If these packages are not operating correctly, then validation is not attempted.

An applicant is permitted to substitute the body of package REPORT with an equivalent one if for some reason the original version provided by the ACVC cannot be executed on the target computer. Package REPORT was not modified for this validation.

All support package specifications and bodies were compiled and were demonstrated to be operating correctly.

2.2- Withdrawn Tests

Some tests are withdrawn from the ACVC because they do not conform to the Ada Standard. When testing was performed, the following 63 tests had been withdrawn for the reasons indicated:

B38105B-AB:
This test requires a specific interpretation of the Ada Standard regarding whether an incomplete type can have discriminant constraints *before* the full type declaration; this interpretation is not fully supported by the Ada Standard or Language Maintenance Committee (LMC).

C45521A..Y-B (25 tests):
Cases C and I define the model interval for the result too narrowly.

C48005C-B:
Lines 38 and 63 of this test should check that the value of the designated object is null.

C48006B-B:
This test requires a specific interpretation of the Ada Standard regarding whether an incomplete type can have discriminant constraints *before* the full type declaration ; this interpretation is not fully supported by the Ada Standard or Language Maintenance Committee.

C64103C-B:
This test should raise CONSTRAINT_ERROR during the conversion at line 179.

C64103D-B:
This test involves a CONSTRAINT_ERROR vs. NUMERIC_ERROR issue that is to be resolved by the Language Maintenance Committee.

C64105E-AB:
For case E, ensure that non-null dimensions of formal and actual parameters belong to both index subtypes (see AI-00313).

C64105F-AB:
For case E, ensure that non-null dimensions of formal and actual parameters belong to both index subtypes (see AI-00313).

B66001A-B:
This test checks (in section G) that a function without parameters, which is equivalent to an enumeration literal in the same declarative region, is a redeclaration and as such is forbidden. According to the Ada Standard 8.3(17), the explicit declaration of such a function is allowed if an enumeration literal is considered to be an implicitly declared predefined operation. The Ada Standard is not clear on this point. This issue has been referred to the Language Maintenance Committee for resolution. Since the issue cannot be resolved at this time, the test is withdrawn from Version 1.6.

B67001A-B:
Line 414 is missing the "BEGIN NULL; END;" needed to complete the block beginning at line 389 (case H).

B67004A-B:
This default name for a formal generic equality function should not be allowed to be "/=" unless an expanded name is used.

B74103F-B:
This test hinges on whether or not a generic formal type declaration declares a type. This matter will be debated by the Language Maintenance Committee in November.

B74207A-B:
This test requires a specific interpretation of the Ada Standard regarding whether an incomplete type can have discriminant constraints before the full type declaration; this interpretation is not fully supported by the Ada Standard or Language Maintenance Committee.

C93005B-B, C93005C-B:
These tests contain a declaration of an integer variable whose initialization is solely for the purpose of raising an exception. Some compilers will not raise this exception due to their optimization.

C93007B-B:
This test should check for PROGRAM_ERROR rather than TASKING_ERROR (SEE AI-000149).

CA1003B-AB:
A compilation that contains an illegal compilation unit may now be rejected as a whole (see AI-00255/05).

CA1011A*-B:
The test objective should be reversed to be consistent with AI-00199.

CA1108A-B:
A pragma ELABORATE is needed for OTHER_PKG at line 25.

CA1108B-B:
A pragma ELABORATE is needed for FIRST-PKG at line 39 and for LATER-PKG at line 49.

CA2009B-B:
The repetition of the main procedure after the subunit body makes the subunit body obsolete; therefore, an attempt to execute the main procedure will fail.

CA2009E-B:
The repetition of the main procedure after the subunit body makes the subunit body obsolete; therefore, an attempt to execute the main procedure will fail.

CA2009F*-B:
The file CA2009F2-B is missing from this test suite.

BC1013A-B:
The declaration of equality in lines 86-87 is illegal because the parameter type T declared in line 11 is not a limited type (Ada Standard 6.7-4).

BC3204A..D-B (4 tests), BC3205A..D-B (4 tests), BC3405B-B:
Instantiations with types that have *default discriminants* are now legal (see AI-00037).

BC3220B-B:
This test assumes that the staticness of instantiated generic parameters follows from the staticness of the actual parameters of the instantiation. This compiler treats *all such instantiated parameters* as non-static. The matter is before the LMC for resolution.

BC3503A-B:
This test requires a specific interpretation of the Ada Standard regarding whether an incomplete type can have discriminant constraints before the full type declaration ; this interpretation *is not fully* supported by the Ada Standard or Language Maintenance Committee.

CE2107E-B:
This test has a variable, TEMP_HAS_TRUE, that needs to be given an initial value of TRUE.

CE3603A-B:
The last case is inconsistent with AI-00050. If string argument is null, no attempt to read is made and END_ERROR is not raised.

CE3604A-B:
Cases 5,8,9, and 11 are inconsistent with AI-00050. SKIP_LINE is *called only if* the end of the output string has not been met.

CE3704M-B:
A *superfluous* SKIP_LINE causes the input and output operations to be out of synchronization.

2.3- Inapplicable Tests

Somes tests use features of the Ada language that the Ada Standard does not require a compiler to support; thus these tests may be inapplicable to a particular compiler. Others may depend on the result of another test that is either inapplicable or withdrawn. For this validation attempt, 275 tests were inapplicable for the reasons indicated:

A91002M-B.ADA:
This test is inapplicable because this implementation does not support certain pragmas such as CONTROLLED.

B86001DT-AB.TST:
This test is inapplicable because this implementation has no predefined type other than INTEGER, FLOAT, SHORT_INTEGER, SHORT_FLOAT, LONG_INTEGER, LONG_FLOAT. The macro name SNAME was set to NO_SUCH_TYPE and the declaration of a procedure name NO_SUCH_TYPE is then legal.

C24113C..Y-B.DEP
C35705C..Y-B.DEP
C35706C..Y-B.DEP
C35707C..Y-B.DEP
C35708C..Y-B.DEP
C35802C..Y-B.DEP
C45241C..Y-B.DEP
C45321C..Y-B.DEP
C45421C..Y-B.DEP
C45424C..Y-B.DEP
C45621C..Z-B.DEP ((10*23)+24=254 tests):
These tests are inapplicable because this implementation limits digits to 6.

B86001CP-AB.DEP
C34001F-B.DEP
C35702A-AB.DEP:
These tests are inapplicable because this implementation does not support SHORT_FLOAT.

B86001CQ-AB.DEP
C34001G-B.DEP
C35702B-AB.DEP:
These tests are inapplicable because this implementation does not support LONG_FLOAT.

B91001G-B.ADA
BC1002A-B.ADA
C55B16A-AB.DEP
C87B62A..C-B.DEP ((1•3)+3 = 6 tests):
These tests are inapplicable because this implementation does not support representation clauses.

C86001F-B.DEP:
This test is inapplicable because this implementation rejects the recompilation of SYSTEM at compilation-time.

CA2009C-B.DEP:
This test is inapplicable because this implementation does not support instantiating missing generic bodies.

LA3004A•-AB.DEP
LA3004B•-B.DEP:
These tests are inapplicable because this implementation does not support pragma INLINE. These tests ignore the pragma and are processed correctly.

C52103X-B.ADA
C52104X-B.ADA
C52104Y-B.ADA:
These tests are inapplicable because this implementation does not support pragma PACK. These tests ignore the pragma and are processed correctly.

2.4 Implementation Characteristics

One of the purposes of validation is to determine the behavior of a compiler in those areas of the Ada Standard that permit implementations to differ. Class D and E tests specifically check for such implementation differences. However, inapplicable tests in other classes also characterize an implementation. This compiler is characterized by the following interpretations of the Ada Standard:

. Non-graphic characters.

Non-graphic characters are defined in the ASCII character set but are not permitted in Ada programs, even within character strings. The compiler correctly recognizes these characters as illegal in Ada compilations. The characters are not printed in the output listing.

. Capacities.

The compiler correctly processes compilations containing loop statements nested to 65 levels, block statements nested to 65 levels, procedures nested to 10 levels, and 723 variables.

. Universal integer calculations.

An implementation is allowed to reject universal integer calculations having values that exceed SYSTEM.MAX_INT. This implementation does not reject such calculations and processes them correctly.

. Universal real calculations.

An implementation is allowed to reject universal real calculations having values that exceed certain precisions. This implementation does not reject such calculations and processes them correctly.

No rounding in this compiler. The precision is arbitrarily high.

. Predefined types.

This implementation supports the predefined types SHORT_INTEGER, LONG_INTEGER, INTEGER, FLOAT, DURATION. It does not support any other predefined numeric types.

41

. Based literals.

An implementation is allowed to reject a based literal with value exceeding SYSTEM.MAX_INT during compilation or it may raise NUMERIC_ERROR during execution. This compiler raises NUMERIC_ERROR during execution.

. Array types.

An implementation is allowed to raise NUMERIC_ERROR for an array having a 'LENGTH that exceeds STANDARD.INTEGER'LAST and/or SYSTEM.MAX_INT. When an array type is declared with an index range exceeding INTEGER values and with a component that is a null BOOLEAN array, this compiler does not raise any exception.

When an array type is declared with an index range exceeding SYSTEM.MAX_INT values and with a component that is a null BOOLEAN array, this compiler raises NUMERIC_ERROR.

A packed BOOLEAN array of length INTEGER'LAST+3 does not raise any exception. A packed two-dimensional BOOLEAN array with INTEGER'LAST+3 components does not raise any exception.

A null array with one dimension of length exceeding INTEGER'LAST does not raise any exception.

In assigning one-dimensional array types, the entire expression is evaluated before CONSTRAINT_ERROR is raised when checking whether the expression's subtype is compatible with the target's subtype. In assigning two-dimensional array types, The entire expression is not evaluated before CONSTRAINT_ERROR is raised when checking whether the expression's subtype is compatible with the target's subtype. In assigning record types with discriminants, the entire expression is evaluated before CONSTRAINT_ERROR is raised when checking whether the expression's subtype is compatible with the target's subtype.

. Discriminated types.

An incompletely declared type with discriminants may be used in an access type definition and constrained either there or in later subtype indications.

. Aggregates.

When evaluating the choices of a multi-dimensional aggregate all choices are evaluated before checking against the index type.

When evaluating an aggregate containing subaggregates, all choices are not evaluated before being checked for identical bounds.

. Functions.

The declaration of a parameterless function with the same profile as an enumeration literal in the same immediate scope is rejected by the implementation.

. Representation clauses.

'SMALL length clauses are not supported.

Enumeration representation clauses are not supported.

. Tasks.

A task object's storage size is not allowed to change after the task is activated.

. Generics.

When given a separately compiled generic declaration, some illegal instantiations, and a body, the compiler rejects the body because of the instantiations.

. Package CALENDAR.

TIME_OF and SPLIT are inverses when SECONDS is a non-model number.

. Pragmas.

Pragma INLINE is not supported for procedures. It is not supported for functions.

. Input/output.

Package SEQUENTIAL_IO can be instantiated with unconstrained array types and record types with discriminants. Package DIRECT_IO can be instantiated with unconstrained array types and record types with discriminants without defaults.

For SEQUENTIAL_IO, DIRECT_IO and TEXT_IO more than one internal file can be associated with each external file for both reading and writing. An external file associated with more than one internal file can be deleted.

An existing text file can be opened in OUT_FILE mode, can be created in OUT_FILE mode, and can be created in IN_FILE mode.

Dynamic creation and resetting of a sequential file is allowed.

Temporary sequential files are given a name. Temporary direct files are given a name. Temporary files given names are deleted when they are closed.

CHAPTER 3

COMPILER ANOMALIES AND NONCONFORMANCES

3.1- Anomalies

An anomaly is a test result that, given the pre-validation analysis, was not expected during formal validation but which is judged allowable by the AVF and the AVO under the circumstances of the validation. No anomalies were detected in this validation attempt.

3.2- Nonconformances

Any discrepancy between expected test results and actual test results is considered to be a nonconformance. No nonconformances were detected in this validation attempt.

CHAPTER 4

ADDITIONAL TESTING INFORMATION

4.1- Pre-Validation

Prior to validation, a set of test results for ACVC 1.6 produced by AlsyCOMP_001, version 1.3 was submitted to BNI by the applicant for pre-validation review. Analysis of these results demonstrated that the compiler successfully passed all applicable tests.

4.2- Test Site

Tests were compiled and executed at Alsys premises in La Celle Saint Cloud - France.

4.3- Test Tape Information

A test tape containing ACVC Version 1.6 was taken on-site by the validation team. This tape contained all tests applicable to this validation as well as all tests inapplicable to this validation except for any Class C tests that require floating-point precision exceeding the maximum value supported by the implementation. Tests that were withdrawn from ACVC 1.6 were not written to the tape. Tests that make use of values that are specific to an implementation were customized before being written to the tape. Any split tests were also included on the test tape so that no editing of the test files was necessary when the validation team arrived on-site.

The test files were mounted on the VAX. Only one directory was used. The format of there test tape was the same as the ACVC distribution tapes.

4.4- Testing Logistics

Once all tests had been loaded to disk, processing was begun using command scripts provided by ALSYS. The text of these scripts are given in Appendix C.

The output of the host machine was on tape. It was then transfered to the target disk using a standard communication line. The operation of loading the target and executing the tests did not depend on the host. This is due to the fact that each machine has its own operating system. The results of execution were transfered back to the VAX to be forwarded to the BNI on tape for analysis.

The compiler supports various options that control its operation. The compiler was tested with the following option settings.

The following options were used :

error_limit=999 : extension of the implicit number of errors before abortion

line=120 : line length

short : no compilation listing

long : compilation listing

banner : banner for each test

nosummary : no recapitulation of errors

The B tests were compiled with the options: error_limit=999, line=120, long, banner, nosummary.

The other tests that do not execute were compiled with the options: error_limit=999, line=120, long, banner, nosummary.

The tests that do execute were compiled with the options: error_limit=999, line=120, short, banner, nosummary.

The tests were run in the following order : A, B, C, D, E and L.

One Ada library per ACVC chapter was used.

4.5- Testing Duration

The ACVC has not been designed for use in measuring compiler performance. The information reported here thus merely describes the duration of the on-site testing for conformity, and is not necessarily an indication of the subject system's performance.

The validation started on the 31 October 1985. It finished on the 8 November 1985. No precise timing information could be collected as numerous problems arised in transfering the files from one machine to the other. This had for consequence several days delay.

CHAPTER 5

SUMMARY AND CONCLUSIONS

The BNI identified 1845 of the 2162 tests in ACVC version 1.6 to be processed during the validation of AlsyCOMP_001, version1.0. Excluded were 254 tests requiring too great a floating-point precision, and the 63 withdrawn tests. 21 tests were determined to be inapplicable after they were processed. The remaining 1824 tests were passed by the compiler.

The BNI concludes that these results demonstrate acceptable conformance to the Ada Standard.

APPENDIX A

COMPLIANCE STATEMENT

The only allowed implementation dependencies correspond to implementation-dependent pragmas and attributes, to certain machine-dependent conventions as mentioned in Chapter 13 of MIL-STD-1815A, and to certain allowed restrictions on representation classes. The implementation-dependent characteristics of the AlsyCOMP_001, version 1.3 are described in the following sections which discuss topics one through eight as stated in Appendix F of the Ada Standard.

(1) Implementation-Dependent Pragmas

None.

(2) Implementation-Dependent Attributes

None.

(3) Package SYSTEM

The specification for package SYSTEM is

```
package SYSTEM is

    type ADDRESS is private;
    type NAME is ( UNIX );

    SYSTEM_NAME  : constant NAME := UNIX;
    STORAGE_UNIT : constant := 8;
    MEMORY_SIZE  : constant := 2**24 - 1;

    -- System-Dependent Named Numbers:

    MIN_INT      : constant := -(2**31);
    MAX_INT      : constant := 2**31-1;
    MAX_DIGITS   : constant := 6;
    MAX_MANTISSA : constant := 31;
    FINE_DELTA   : constant := 2#1.0#e-31;
    TICK         : constant := 1.0;

    -- Other System-Dependent Declarations

    subtype PRIORITY is INTEGER range 1..127;

end SYSTEM;
```

(4) Representation Clause Restrictions

Representation clauses specify how the types of the language are to be mapped onto the underlying machine. The following are restrictions on representation clauses.

Address Clause

Not accepted

Length Clause

Not accepted

Enumeration Representation Clause

Not accepted

Record Representation Clause

Not accepted

(5) Conventions

No implementation-generated names.

(6) Address Clauses

Not accepted.

(7) Unchecked Conversions

The following are restrictions on unchecked conversions, including those depending on the respective sizes of objects of the source and target.

They should have the same size.

(8) Input-Output Packages

The following are implementation-dependent characteristics of the input-output packages.

SEQUENTIAL_IO Package

Declare file type and applicable operations for files of this type.

There is no restriction in the use of sequential Input/Output.

DIRECT_IO Package

```
type COUNT is range 0 .. 2_147_483_647;
```

TEXT_IO Package

```
type COUNT is range 0 .. 2_147_483_647;

subtype FIELD is INTEGER range 0 .. 255;
```

(9) Package STANDARD

```
type INTEGER is range -32768 .. 32767;
type SHORT_INTEGER is range -128 .. 127;
type LONG_INTEGER is -2_147_483_648.. 2_147_483_647;

type FLOAT is digits 6 range
                -2#1.111_1111_1111_1111_1111_1111#E+127
             .. 2#1.111_1111_1111_1111_1111_1111#E+127;
```

No other additional predefined floating point types

```
type DURATION is delta 0.002 range -86_400.0 .. 86_400.0;
```

No other predefined types

(10) File Names

File names make no use of conventions except those of the operating system.

APPENDIX B

TEST PARAMETERS

Certain tests in the ACVC make use of implementation-dependent values, such as the maximum length of an input line and invalid file names. A test that makes use of such values is identified by the extension .TST in its file name. Actual values to be substituted are identified by names that begin with a dollar sign. A value is substituted for each of these names before the test is run. The values used for this validation are given below.

Name and Meaning	Value
$MAX_IN_LEN Maximum input line length permitted by the implementation.	255
$BIG_ID1 Identifier of size MAX_IN_LEN with varying last character.	X2345678901234567890123456789012345 6789012345678901234 5AAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAA1
$BIG_ID2 Identifier of size MAX_IN_LEN with varying last character.	X2345678901234567890123456789012345 6789012345678901234 5AAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAA2

Name and Meaning	Value
$BIG_ID3 Identifier of size MAX_IN_LEN with varying middle character.	X2345678901234567890123456789012345 67890123456789012345AAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAA3AAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAA
$BIG_ID4 Identifier of size MAX_IN_LEN with varying middle character.	X2345678901234567890123456789012345 67890123456789012345AAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAA4AAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAA
$NEG_BASED_INT A based integer literal whose highest order non-zero bit falls in the sign bit position of the representation for SYSTEM.MAX_INT.	16#FF_FF_FF_FD#
$BIG_INT_LIT An integer literal of value 298 with enough leading zeroes so that it is MAX_IN_LEN characters long.	00000000000000000000000000000000000 00000000000000000000000000000000000 00000000000000000000000000000000000 00000000000000000000000000000000000 00000000000000000000000000000000000 00000000000000000000000000000000000 00000000000000000000000000000000000 0000000298
$BIG_REAL_LIT A real literal that can be either of floating or fixed point type, has value 690.0, and has enough leading zeroes to be MAX_IN_LEN characters long.	00000000000000000000000000000000000 00000000000000000000000000000000000 00000000000000000000000000000000000 00000000000000000000000000000000000 00000000000000000000000000000000000 00000000000000000000000000000000000 00000000000000000000000000000000000 000069.0E1
$EXTENDED_ASCII_CHARS A string literal containing all the ASCII characters with printable graphics that are not in the basic 55 Ada character set.	"abcdefghijklmnopqrstuvwxyz!$%?@[\]" & "†'{}"

Name and Meaning	Value
$NON_ASCII_CHAR_TYPE An enumerated type definition for a character type whose literals are the identifier NON_NULL and all non-ASCII characters with printable graphics.	(NON_NULL)
$BLANKS Blanks of length MAX_IN_LEN - 20	
$MAX_DIGITS Maximum digits supported for floating point types.	6
$NAME A name of a predefined numeric type other than FLOAT, INTEGER, SHORT_FLOAT, SHORT_INTEGER, LONG_FLOAT, LONG_INTEGER, or DURATION.	NO_SUCH_TYPE
$INTEGER_FIRST The universal integer literal expression whose value is INTEGER'FIRST.	-32768
$INTEGER_LAST The universal integer literal expression whose value is INTEGER'LAST.	32767
$LESS_THAN_DURATION A universal real value that lies between DURATION'BASE'FIRST and DURATION'FIRST or any value in the range of DURATION.	-100_000.0
$GREATER_THAN_DURATION A universal real value that lies between DURATION'BASE'LAST and DURATION'LAST or any value in the range of DURATION.	100_000.0

Name and Meaning	Value
$LESS_THAN_DURATION_BASE_FIRST The universal real value that is less than DURATION'BASE'FIRST.	-100_000_000.0
$GREATER_THAN_DURATION_BASE_LAST The universal real value that is greater than DURATION'BASE'LAST.	100_000_000.0
$COUNT_LAST Value of COUNT'LAST in TEXT_IO package.	2_147_483_647
$FIELD_LAST Value of FIELD'LAST in TEXT_IO package.	255
$FILE_NAME_WITH_BAD_CHARS An illegal external file name that either contains invalid characters or is too long.	ABCDEFGHIJKLMNOPQRSTUVWXYZ
$FILE_NAME_WITH_WILD_CARD_CHAR An external file name that either contains a wild card character or is too long.	123456789012345
$ILLEGAL_EXTERNAL_FILE_NAME1 Illegal external file name.	BAD_CHARACTER*†
$ILLEGAL_EXTERNAL_FILE_NAME2 Illegal external file name.	MUCH-TOO-LONG-NAME-FOR-A-FILE

APPENDIX C

COMMAND SCRIPTS

```
_DUA1:[PRODUCTIZATION.COMMAND]SEND_OBJ.BAT;20                                  5-NO
$!*****************************************************************************
$!    Send object files to a target computer as soon as created.
$!
$! syntax: submit send_obj.bat /param=(product_name)
$!
$! version 1 by CDC, 10-FEB-1989
$! version 2 by MS    07-AUG-1989
$!
$!*****************************************************************************
$!
$! Define the product and the logical names defined for the qualification:
$!
$  @user1:[productization]login.com
$  @tools:setproduct 'p1'
$!
$  set mess/not/ext/nohd/nofac/nosev
$!
$  product_version_name = alsys_product
$  product_name         = f$parse(alsys_product,,,"name")
$  define product_name    'product_name'
$  product_version      = f$parse(alsys_product,,,"version")
$  version              = product_version - "j"
$!
$  if p2 then goto debug     ! p3 = y
$!
$! not debug
$     @dcom:setup
$     define product_domain -
       user1:[productization.products.'product_name'.v'version'.qualification.]
$     define temp qualif0:[temp]
$     define line_to_target   txb3:
$     define line_from_target txa7:
$     goto start
$!
$  debug:
$     @dcom:setup
$     define product_domain -
        user1:[productization.products.'product_name'.v'version'.qualif_debug.]
$     define temp  qd0:[temp]
$     define line_to_target   txb2:
$     define line_from_target txa6:
$!
$  start:
$!
$ set noon
$ proc_name = "Send "+f$trnlnm("line_to_target")
$ set proc /name="'proc_name'"
$ set on
$!
$ kermit := $exe:kermit.exe
$ iddle = 0
$ on error then goto cleanup
$!
$ define kerScomm line_to_target
$ alloc line_to_target
$ set term/hos*sync/noreadsync/type=vax/unknown/speed=9600 line_to_target
$!
$inop:
```

```
$   file = f$search("temp:*.o")
$   if file .eqs. "" then goto wait
$retry:
$   open /read/err=locked obj_file 'file'
$   close obj_file
$   idle = 0
$   set noon
$   kermit send 'file'
$   del 'file'
$   set on
$   goto loop
$!
$wait:
$   if idle .eq. 240 then goto failed  ! don't wait more than 4 hours
$   idle = idle + 1
$   wait 0:01 ! wait one minute.
$   goto loop
$!
$locked:
$   wait 0:01
$   goto retry
$!
$failed:
$   write sys$output "No object file found after 2 hours"
$cleanup:
$   on error then exit
$   kermit logout
$   dealloc line_to_target
$!
$!-------------------- That's all ----------------------------------------
```

```
$!*****************************************************************************
$!             Start a complete qualification train
$! syntax: @start_qualif [list [product [traces [options]]]]
$! submits 3 batch jobs :
$!    . the qualification job : - qualif_train.bat
$!    . the transmission jobs : - to send the objects to the target
$!                              - to receive the results from the target
$! Six batch queues are necessary:
$!    . qualif$batch   )
$!    . qualif$debug   )   for the qualification batches
$!    . send$batch     )
$!    . send$debug     )   for the send batches
$!    . receive$batch  )
$!    . receive$debug  )   for the receive results batches
$! These batch queues must only have one entry each.
$!*****************************************************************************
$!
$  default_product       = "001;2"
$  default_traces        = "690"
$  default_options       = "/LONG/NOSU"/BANNER/LINE=120/BROWSE=999"
$  default_cpu           = 60
$  default_debug         = "n"
$  default_bind_traces   = ""
$  default_qualification = "n"
$!
$ if p1 .eqs. "" then inquire p1 "List file"
$ p1 = f$parse(p1,".lst")
$ if f$search("''p1'") .nes. "" then goto check_p2
$ write sys$output "*** File ''p1' not found"
$ exit
$!
$check_p2:
$  if p2 .eqs. "" then inquire p2 "Product_name (''default_product')"
$  if p2 .eqs. "" then p2 = default_product
$  @tools:setproduct 'p2'
$!
$  qualification = p3
$  if p3 .eqs. "" then inquire -
          qualification "Qualification (''default_qualification')"
$  if qualification .eqs. "" then qualification = default_qualification
$!
$  if p4 .eqs. "" then inquire debug "Debug option (''default_debug')"
$  if debug .eqs. "" then debug = default_debug
$!
$  traces = p5
$  if traces .eqs. "" then inquire traces "Compile traces (''default_traces')"
$  if traces .eqs. "" then traces = default_traces
$!
$  if p3 then goto qualif
$!
$! validation
$!
$  options     = default_options
$  cpu_time    = default_cpu
$!
$  default_bind_traces = 445
$  inquire bind_traces "Bind traces (''default_bind_traces')
$  if bind_traces .eqs. "" then bind_traces = default_bind_traces
```

```
$!
$  goto both
$!
$ qualif:
$!
$  options = ""
$  if options .eqs. "" then inquire options "uctions (''default_options')"
$  if options .eqs. "" then options = default_options
$!
$  cpu_time = 07
$  if cpu_time .eqs. "" then inquire cpu_time "Max cpu_time (''default_cpu')"
$  if cpu_time .eqs. "" then cpu_time = default_cpu
$!
$  inquire bind_traces "Bind traces (''default_bind_traces')
$  if bind_traces .eqs. "" then bind_traces = default_bind_traces
$!
$ both:
$!
$ set mess /nofac/nosev/notext/noid
$!
$  if debug then goto dd
$!
$! not debug
$!
$      send_queue    = "send$batch"
$      receive_queue = "receive$batch"
$      qualif_queue  = "qualif$batch"
$      define temp     "''f$trnlnm("qtmp")'"
$     goto begin
$!
$ dd:
$!
$      send_queue    = "send$debug"
$      receive_queue = "receive$debug"
$      qualif_queue  = "qualif$debug"
$      define temp     "''f$trnlnm("qdtmp")'"
$!
$ begin:
$!
$ listname = f$parse ("''p1'",,,"NAME")
$!
$ set mess /fac/sev/text/id
$!
$ submit/keep/noprint                    -
        /log=temp:S_'listname'.log       -
        /param=("''p2'","''debug'")      -
        /name=S_'listname'-
        /queue="''send_queue'"           -
             pcom:send_obj.bat
$!
$ submit/keep/nopr$nt                          -
        /log=temp:b_'listname'.log       -
        /param=("''p2'","''p1'","''debug'","''qualification'")  -
        /name=?_'listname'-
        /queue="''receive_queue'"                -
             pcom:receive_results.bat
$!
$ submit/keep/noprint/noti'y             -
```

```
_DUA1:[PRODUCTION.COM]DISTANT_QUALIF.COM;49                                  29-

        /log=temp:"_'listname'.log     -
        /param("'p1'","'p2'","'traces'","'options'", -
               "'cpu_time'","'debug'","'p3'","'bind_traces'") -
        /name="_'listname'"-
        /queue="'qualif_queue'"       -
             @com:qualif_train_nodif.bat
$!
$!-------------------------------------------------------------------------q
```

```
$ @VAL:[PRODUCTIZATION.COMMANDS]START_QUALIF_NOPIF.COM;7

$!*****************************************************************************
$!              Start a complete qualification train
$! syntax: @start_qualif [list [product [traces [options]]]]
$! submits 3 batch jobs :
$!     . the qualification job : - qualif_train.bat
$!     . the transmission jobs : - to send the objects to the target
$!                               - to receive the results from the target
$! Six batch queues are necessary:
$!     . qualif$batch   )
$!     . qualif$debug   )   for the qualification batches
$!     . send$batch     )
$!     . send$debug     )   for the send batches
$!     . receive$batch  )
$!     . receive$debug  )   for the receive results batches
$! These batch queues must only have one entry each.
$!*****************************************************************************
$!
$  default_product        = "001;2"
$  default_traces         = "640"
$  default_options        = "/LONG/""USE""/NUMBER/LINE=120/ERROR=999"
$  default_cpu            = 60
$  default_debug          = "n"
$  default_bind_traces    = ""
$  default_qualification  = "n"
$!
$ if p1 .eqs. "" then inquire p1 "List file"
$ p1 = f$parse(p1,".lst")
$ if f$search("''p1'") .nes. "" then goto check_p2
$ write sys$output "*** File ''p1' not found"
$ exit
$!
$check_p2:
$  if p2 .eqs. "" then inquire p2 "Product_name (''default_product')"
$  if p2 .eqs. "" then p2 = default_product
$  @tools:setproduct 'p2'
$!
$  qualification = p3
$  if p3 .eqs. "" then inquire -
          qualification "Qualification (''default_qualification')"
$  if qualification .eqs. "" then qualification = default_qualification
$!
$  if p4 .eqs. "" then inquire debug "Debug option (''default_debug')"
$  if debug .eqs. "" then debug = default_debug
$!
$  traces = p5
$  if traces .eqs. "" then inquire traces "Compile traces (''default_traces')"
$  if traces .eqs. "" then traces = default_traces
$!
$  if p3 then goto qualif
$!
$! validation
$!
$  options     = default_options
$  cpu_time    = default_cpu
$!
$  default_bind_traces = 445
$  inquire bind_traces "Bind traces (''default_bind_traces')
$  if bind_traces .eqs. "" then bind_traces = default_bind_traces
```

```
_DUA1:[PRODUCTION.COMMAND]START_QUALIF_MODIF.COM;7

$!
$  goto both
$!
$ qualif:
$!
$  options = p6
$  if options .eqs. "" then inquire options "options (''default_options')"
$  if options .eqs. "" then options = default_options
$!
$  cpu_time = p7
$  if cpu_time .eqs. "" then inquire cpu_time "Max cpu_time (''default_cpu')"
$  if cpu_time .eqs. "" then cpu_time = default_cpu
$!
$  inquire bind_traces "Bind traces (''default_bind_traces')
$  if bind_traces .eqs. "" then bind_traces = default_bind_traces
$!
$ both:
$!
$ set mess /nofac/nosev/notext/noid
$!
$  if debug then goto dd
$!
$! not debug
$!
$      send_queue    = "send$batch"
$      receive_queue = "receive$batch"
$      qualif_queue  = "qualif$batch"
$      define temp     "''f$trnlnm("qtmp")'"
$     goto begin
$!
$ dd:
$!
$      send_queue    = "send$debug"
$      receive_queue = "receive$debug"
$      qualif_queue  = "qualif$debug"
$      define temp     "''f$trnlnm("qdtmp")'"
$!
$ begin:
$!
$ listname = f$parse ("''p1'",,,"NAME")
$!
$ set mess /fac/sev/text/id
$!
$ submit/keep/noprint                    -
        /log=temp:S_'listname'.log       -
        /param=("''p2'","''debug'")      -
        /name=S_'listname'-
        /queue="''send_queue'"           -
             pcom:send_obj.bat
$!
$!
$ submit/keep/noprint/notify             -
        /log=temp:S_'listname'.log       -
        /param=("''p1'","''p2'","''traces'","''options'", -
               "''cpu_time'","''debug'","''p3'","''bind_traces'") -
        /name=S_'listname'-
        /queue="''qualif_queue'"         -
             pcom:qualif_train_nouif.bat
```

```
_DUA1:[PRODUCTIZATION.COMMANDS]START_QUALIF_MODIF.COM;7

$!
$!-------------------------------------------------------------------------------
```

```
_DUA1:[ADA.ACVC.QUALIF_TRAIN_MODIF.COM]1                                    6-NOV-1986

$!*****************************************************************************
$! Train de compilation et de bind pour la qualification d'un compilateur.
$!
$! Cette commande traite les tests de classe A,B,C,D,E,I,K et L.
$!      A : tests corrects, doivent seulement etre compiles, pas bindes.
$!      B : tests deviants, ils contiennent des erreurs, pas de bind.
$!      C : tests executables, doivent etre compiles et bindes avec succes.
$!      D : tests de performance
$!      E : comme C mais specifique d'une implementation
$!      I : tests interactifs
$!      K : tests d'information
$!      L : tests contenant une erreur devant etre detectee au bind.
$!
$!      Pour etre valide le compilateur doit fournir les resultats suivants
$!      pour tous les tests qui lui sont soumis, de plus les erreurs detectees
$!      doivent etre celles attendues
$!
$!      Classe  Compilation     Bind    Execution
$!      ---------------------------------------
$!         B       failed        --        --
$!         A         OK          --        --
$!         L         OK        failed      --
$!         C         OK          OK        OK
$!         E         OK          OK        OK
$!         I         OK          OK        OK   ( execution non automatique )
$!
$! Les tests de classes D et K n'interviennent pas directement dans le
$! processus de validation.
$!
$! version 2 modifie par CBC, 10-JAN-1985
$! version 3, MS 08-07-85
$!
$!*****************************************************************************
$!
$! syntax: submit qualif_train.dat -
$! /param=(list, product_name, traces, options, max_cpu, debug or not,
$!         qualification or validation )
$!
$!*****************************************************************************
$!
$! Define the product and the logical names defined for the qualification:
$!
$  @user1:[productization]login.com
$  @tools:setproduct 'p2'
$!
$  set mess/no*ext/noid/nofac/nosev
$!                                                                            !
$  kermit :== $user1:[productization.commands]kermit.exe                      !
$  product_version_name = alsys_product                                       !
$  product_name         = f$parse(alsys_product,,,"name")                     !
$  product_version      = f$parse(alsys_product,,,"version")                  !
$  version              = product_version - ";"
$!
$  define product_name    'product_name'
$  define tests           user1:[productization.tests.]
$!
$  edit_base   :== $user1:[productization.commands]edit_base.exe
$  create_base :== $user1:[productization.commands]create_base.exe
```

```
$  update_base :== $user1:[productization.commands]update_base.exe
$!
$  if p6 then goto debug     ! p6 = y
$!
$! not debug
$     @dcom:setup
$     define product_domain -
        user1:[productization.products.'product_name'.v'version'.qualification.]
$     define temp     -
  user1:[productization.products.'product_name'.v'version'.qualification.temp]
$     define resu     -
 user1:[productization.products.'product_name'.v'version'.qualification.results]
$     define adalib   -
 user1:[productization.products.'product_name'.v'version'.qualification.train_ac
$     define reinit_adalib  -
 user1:[productization.products.'product_name'.v'version'.qualification.acvc_ada
$     define base     -
 user1:[productization.products.'product_name'.v'version'.qualification.status]
$!
$     define line_to_target   txb3:
$     define line_from_target txa7:
$!
$     goto start
$!
$  debug:
$     @dcom:setup
$     define product_domain -
          user1:[productization.products.'product_name'.v'version'.qualif_debug.]
$     define temp     -
              user1:[productization.products.'product_name'.v'version'.qualif_deb
$     define resu     -
  user1:[productization.products.'product_name'.v'version'.qualif_debug.results]
$     define adalib   -
  user1:[productization.products.'product_name'.v'version'.qualif_debug.train_ad
$     define reinit_adalib  -
  user1:[productization.products.'product_name'.v'version'.qualif_debug.acvc_ada
$     define base     -
 user1:[productization.products.'product_name'.v'version'.qualification.status]
$!
$     define line_to_target   txb2:
$     define line_from_target txa5:
$!
$  start:
$   set mess/text/id/fac/sev
$   ada68k set adalib
$!
$   write sysoutput "---- Log file for the list ''p1'"
$!
$   report := write report_file
$!
$   object_dir := product_domain:[temp]
$!
$! Miscellaneous initializations
$      ! directory of files to process
$   dir_name = f$trnlnm("tests:")
$      ! counters
$   nb_total   = 0
$   nb_invalid = 0
```

```
$   no_check    = 0
$   err_level   = 1 ! success value
$      ! valid options and test classes
$   valid_options = "/PRINT/TRACE/OPT/LIB/DIR/BIN"/CPU/PRINT"
$   valid_class   = "ABCDEIKL"
$      ! initialize cpu limit,traces and options
$!
$   max_cpu       = "'p5'"          ! maximum cpu time in minutes
$   traces        = "'p3'"
$   options       = "'p4'"
$   debug_option  = "'p6'"
$   qualification = "'p7'"
$   bind_traces   = "'p8'"
$!
$      ! separators
$   short_separ= "----------------------------------------"
$   separ = short_separ + short_separ
$   killed = ""0"
$!
$! Open all the files
$   list_name = "product_domain:[results]" + f$parse(p1,,,"NAME")
$!
$   !1. the list file with the complete name of each test
$   open/read list_file 'p1'
$!
$   !2. the report file that contains the result of each compilation
$   rpt_name = list_name + ".rpt"
$   open/write report_file 'rpt_name'
$   rpt_name = f$search(rpt_name) ! get full name, including version number
$!
$   !3. the check file that contains the names of the programs for wich the
$   !   compilation failed and a manual check is required.
$   check_name = list_name + ".chk"
$   open/write  check_file  'check_name'
$!
$   !4. the error file wich contains all the compilation errors
$   error_name = list_name + ".err"
$   create 'error_name'
$   error_name = f$search(error_name) ! get full name
$   open/append error_file 'error_name'
$!
$! Write a header for the report file and the error file
$!
$   line = f$fao("!50AS !AS",alsys_product, f$time())
$   report line
$   write error_file  line
$   report ""
$   write error_file  ""
$   report "Test report for ", p1
$   write error_file  "Error file for ",p1
$   report ""
$   write error_file  ""
$   report separ
$   write error_file  separ
$   report ""
$   write error_file  ""
$   line = f$fao -
       ("!33AS !5*S !11AS !11AS !AS",-
```

```
        "File name","Class","Compilation","    bind","Validated")
$   report line
$   report ""
$!
$   report "-- options       = ",options
$   report "-- max_cpu       = ",max_cpu
$   report "-- traces        = ",traces
$   report "-- debug         = ",debug_option
$   report "-- qualification = ",qualification
$   report "-- bind_traces   = ",bind_traces
$   report " "
$!
$!*****************************************************************************
$!                              the main loop
$!*****************************************************************************
$!
$   write sys$output "----- Starting main loop ----"
$!
$loop:
$   read/end=finish list_file input_line        ! read one record
$   !
$   if f$extract(0,1,input_line) .eqs. "$" then goto dcl_command
$   input_line := 'input_line'  ! remove comments and switch to upper case
$   if input_line .eqs. "" then goto loop
$   if f$extract(1,1,input_line) .nes. " " then goto not_a_test
$   !
$   ! Here is a source file
$   !
$   class = f$extract(0,1,input_line)
$   test_class = f$extract(2,3,input_line)
$   if f$locate(class,valid_class) .eq. f$length(valid_class) then -
        goto bad_command
$   nb_total = nb_total + 1             ! count the tests
$   filename = f$extract(2,99,input_line)
$   main_program = f$parse(filename,,,"NAME")
$   if main_program .eqs. "" then goto bad_command
$   filename = main_program + f$parse(filename,".ADA",,"TYPE")
$   source =  dir_name + filename
$   !
$   write sys$output "Treating ''main_program'"
$   !
$   ! Allow reading of these files while running
$   close report_file
$   close error_file
$   !
$   ! Run the Compiler
$   !-----------------
$   set noon
$   set mess /notext/nosev/nofac/noid
$   proc_name = "A " + f$extract(0,13,'filename) ! A for Ada
$   set process/name="''proc_name'"
$!
$   set on
$   set mess /text/nosev/nofac/noid
$   on error then goto err_test
$!
$   @pcom:compile_it.bat "''source'" "''options'" -
                         "''traces'" "''max_cpu'" -
```

```
                        "''debug_option'" "''qualification'"
$!      the default is the default one given in the add68k set command
$   !
$   err_test:   ! Compiler error handling
$   err_level = f$integer(f$trnlnm("err_level","LNM$JOB"))
$   killed = f$trnlnm("killed","LNM$JOB")
$   if killed .eqs. "YES" then err_level = 6
$      ! err_level = odd number: success, even number: failure.
$      ! 1 ok, 2 failed, 3 warning, 4 crashed, 6 immediate abort
$   on error then goto unexpected ! avoid loop (on error goto err_test)
$!
$ compiled:
$   open/append/error=retry_for_rpt report_file 'rpt_name'
$   bind_result = "    --"
$   open_err:
$   open/append/error=retry_for_err error_file 'error_name'
$!
$   if .not. qualification then goto compile_result   ! validation => don't
$!                            delete listings and don't write in error_file
$!
$   listing = f$parse ("product_domain:[results].lis",,,filename)
$   set message /notext/nosev/noid/nofac
$   define listing 'listing'
$   set message /text/sev/id/fac
$!
$   if err_level .eq. 1 then goto del_out_text
$!
$! Other cases:
$! The compilation failed or crashed => appends the listing to the error file
$!
$  close error_file
$  set noon
$  append 'listing' 'error_name'
$  set on
$  write error_file ""
$  write error_file "The above error was for ",dir_name,filename
$  write error_file separ
$   !
$   del_out_text:
$      delete 'listing'
$!
$  compile_result:
$!
$! Switch on the compilation result
$!
$   if err_level .eq. 0 then err_level = 3      ! warning
$   if err_level .ne. 1 .and. err_level .ne. 2 .and. err_level .ne. 3 -
      .and. err_level .ne. 4 .and. err_level .ne. 6 then goto unknown_result
$   goto result_'err_level' ! 1, 2, 3, 4 or 6
$   !
$   result_6:
$      comp_result = "* ABORTED"
$      if killed .eqs. "YES" then comp_result = "* LOOPS"
$      goto no_good
$      !
$   result_2:
$      comp_result = "- FAILED"
$!
```

```
$    check_deviant:
$       if class .nes. "B" then goto no_good
$          !
$          ! class = deviant --> listing must be checked manually
$          ! actually, for the validation it is possible to compare
$          ! automatically with a previous set of results
$          !
$       ! validated = "?"
$       ! !
$       ! write check_file dir_name, filename
$!      ! nb_check = nb_check + 1
$          goto write_result
$          !----------------
$!
$    result_4:
$       comp_result = "* CRASHED"
$    !
$    no_good:
$       validated = "NO"
$       nb_invalid = nb_invalid + 1
$       goto write_result
$       !----------------
$    !
$    ! no error in compilation
$    !
$    result_3:
$       comp_result = "  warning"
$       goto check_if_deviant
$       !--------------------
$    !
$    result_1:
$       comp_result = "  success"
$    !
$    check_if_deviant:
$       if class .eqs. "B" then goto no_good
$          ! compare the *.lis files with HP reference
$          validated  := "--"
$          if class .eqs. "A" then goto is_validated
$       if class .eqs. "I" then goto interactif
$       goto do_bind
$       !-----------
$     !
$is_validated:
$          validated = "YES"
$          goto write_result
$          !----------------
$interactif:
$          validated = "--"
$          goto write_result
$          !----------------
$       !
$       do_bind:
$          object_name  = main_program + ".O"
$          goto binds_it
$          !------------
$    unknown_result:
$        report "-- Unknown error level : ". err_level
$        comp_result = " ??????"
```

```
$       validated = "NO"
$!
$   write_result:
$       line = f$fao("!3AS !3AS !11AS !11AS !AS",-
                filename,class,comp_result,bind_result,validated)
$       report line
$!      if ( err_level .eq. 4 ) .or. ( err_level .eq. 6 ) then goto case_rei
$       goto loop
$       !--------
$!
$not_a_test:
$!==========
$   on error then goto bad_command
$   line_head = f$extract(0,3,input_line)
$   after_equal := 'f$extract(f$locate("=",input_line)+1, 99, input_line)
$!    (Remove also leading blanks after the '=')
$   if f$locate("/"+line_head,valid_options) .eq. f$length(valid_options) -
       then goto bad_command
$   goto case_'line_head'
$   !
$   case_bin:   ! BIND makes the binder to be invoked
$   !=============================================
$       main_program := 'f$extract(5,99,input_line)
$       space_pos = f$locate (" ", main_program)
$       main_program = f$extract (0,space_pos,main_program)
$       class= " "
$       comp_result = " "
$!
$       !
$       ! don't bind if previous compilation crashed (but allow failures
$       ! if there are multiple units in a single source file):
$       if err_level .lt. 4 then goto binds_it
$          bind_result = "  not done"
$          err_level = 1                  ! success value
$          goto write_result
$          !----------------
$       binds_it:
$          set noon
$          set mess /no*ext/nosev/nofac/noid
$          proc_name = "6 " + f$extract(0,13,main_program) ! 6 for Bind
$          set process/name="''proc_name'"
$          set on
$          set mess /text/nosev/nofac/noid
$          on error then goto bind_error
$!
$          if bind_traces .eqs. "" then -
              ada68k bind 'main_program' -
                        /in*formational/warning        -
                        /lis=resu:'main_program'.bnd -
                        /out=temp:'main_program'.o
$!
$             ada68k bind 'main_program' -
                        /in*formational/warning        -
                        /lis=resu:'main_program'.bnd -
                        /out=temp:'main_program'.o    -
                        /tra e='bind_traces'
$!
$          ! bind is successful
```


10-2

```
$             bind_result = "  success"
$             if class .eqs. "L" then goto bad_bind ! class L : bind must failed
$                validated = "--" ! it's not finished : execution not yet done
$!               if .not. qualification then goto write_result
$              bind_listing = f$parse("product_domain:[results].bnd",,,filename)
$                set message /nocontext/nosev/noid/nofac
$                define bind_listing 'bind_listing'
$                set message /text/sev/id/fac
$                delete 'bind_listing'
$                goto write_result
$                !---------------
$             bad_bind:
$                validated = "NO"
$                goto write_result
$                !---------------
$          bind_error:
$             bind_result = "= FAILED"
$             if class .eqs. "L" then goto check_bind
$                validated = "NO"
$                goto write_result
$                !---------------
$            check_bind:
$                validated = "?"
$                goto write_result
$                !---------------
$    !
$    case_rei:   ! REINIT causes Ada library reinitialization
$    !=============================================================
$       set mess /text/fac/sev/id
$       ada68k copy reinit_adalib: adalib: /override
$       if .not. $status then goto rei_err
$       purge adalib:
$       report "-- library reinitialized."
$       goto loop
$       !--------
$!
$       rei_err:
$       report "*** library not reinitialized"
$       goto loop
$!
$    case_pri:   ! PRINT = ada68k print adalib
$    !=========================================
$       if .not. debug_option then goto print_ob
$         ada68k print adalib /nopredef/nocol/output=temp:print.lib
$                                    ! print/nopredef/nocollection
$         if .not. $status then goto pri_err
$         report "-- library printed."
$         goto loop
$!
$       print_ob:
$         report "*** print not allowed : not debug compiler."
$         goto loop
$!
$       pri_err:
$         report "*** library not printed"
$         goto loop
$!
$    case_opt:   ! OPT= causes new options to be considered
```

```
$    !=================================================
$       options = after_equal
$       report "-- Options = ",options
$       goto loop
$       !--------
$    !
$    case_tra:    ! TRACE= causes new traces to be considered
$    !=================================================
$       if .not. debug_option then goto trace_db
$       traces = after_equal
$       report "-- Traces = ", traces
$         goto loop
$!
$       trace_db:
$         write sys$output "**** option trace not allowed : not debug compiler."
$         goto loop
$!
$    case_cpu:
$    !========
$       max_cpu = after_equal
$       report "-- Max cpu = ", max_cpu
$       goto loop
$       !--------
$    !
$    case_dir:
$    case_lib:    ! LIB= or DIR= for a new source file directory specification
$    !=================================================================
$       dir_name=f$parse(after_equal,,,"NODE")+ -
                 f$parse(after_equal,,,"DEVICE")+ -
                 f$parse(after_equal,,,"DIRECTORY")
$       report "-- Directory = ",dir_name
$       goto loop
$       !--------
$    !
$    dcl_command:         ! DCL commands start with '$'
$    !=============================================================
$       report "-- ",input_line
$       'f$extract(1,99,input_line)
$       goto loop
$       !--------
$    !
$    bad_command:
$       report "*** Bad command: ", input_line
$       goto loop
$       !--------
$!
$!*************************************************************************
$!                              end of main loop
$!*************************************************************************
$!
$finish:
$!
$! write some statistics in the report file
$    report separ
$    report ""
$    line = f$fao("!AS !5UW","Total number of tests:",nb_total )
$    report line
$!
```

```
$    if nb_total .eq. 0 then goto close_files
$    per_fal = ( nb_invalid * 100 ) / nb_total
$    per_chk = ( nb_check   * 100 ) / nb_total
$    nb_success = nb_total - nb_invalid - nb_check
$    per_suc = 100 - per_fal - per_chk
$!
$    line = f$fao("!12AS !4S !5UW !_(!3UW%)"," "," success:",nb_success,per_suc )
$    report line
$    line = f$fao("!13AS !4S !5UW !_(!3UW%)"," "," failed:",nb_invalid,per_fal )
$    report line
$    line = f$fao("!cAS !4S !5UW !_(!3UW%)"," "," to be checked:",nb_check,per_ch
$    report line
$!
$close_files:
$    close report_file
$    close list_file
$    close error_file
$    close check_file
$!
$    if nb_invalid .eq. 0 then delete 'error_name'
$    if nb_check   .eq. 0 then delete 'check_name';0
$!
$the_end:
$    set noon
$!   if debug_option then exit 1          ! debug_option = y
$!   update_base 'rpt_name' 'product_name'
$    exit 1
$!
$unexpected:
$    report "*** Unexpected error"
$    goto loop
$!
$!*****************************************************************************
$!                          open error handler
$!*****************************************************************************
$!
$open_err1:
$    write sys$output "Unable to open ", p1
$    exit
$!
$open_err2:
$    close list_file
$    write sys$output "Unable to open ", rpt_name
$    exit
$!
$open_err3:
$    close list_file
$    close report_file
$    write sys$output "Unable to open ", check_name
$    exit
$!
$open_err4:
$    close list_file
$    close report_file
$    close check_file
$    write sys$output "Unable to open ", error_name
$    exit
$!
```

```
_DUA1:[APO.100]LIF_TRAIN_MODIF.COM;1                                    6-NOV-198

$retry_for_rd:
$   wait 0:0:30 ! wait 30 seconds
$   goto compiled
$!
$retry_for_err:
$   wait 0:0:30 ! wait 30 seconds
$   goto open_err
$!
$retry_again:
$   wait 0:0:30
$   goto reopen
$!
$!*****************************************************************************
$!                              That's all folks
$!*****************************************************************************
```

```
_DUA1:[PRODUCTIZATION.COMMAND]COMPILE_IT.COM;32                    24-C

$!-------------------------------------------------------------------------
$! Creates a subprocess to run the compiler, limits the cpu time and the
$! size of the log file crash
$!
$! syntax : @compile_it source options [max_cpu]
$!
$! The maximum cpu time must be in minutes. The default is 10 minutes.
$!
$! If the subprocess is killed then the symbol "killed" is set to "YES", else
$! it is set to "NO". The subprocess communicates with the parent process with
$! the logical name "err_level" in the group table.
$!-------------------------------------------------------------------------
$!
$   set noon
$!
$   source         = "'p1'"
$   options        = "'p2'"
$   traces         = "'p3'"
$   debug_option   = "'p5'"
$   qualification  = "'p6'"
$   max_cpu        = 60 * f$integer(p4)
$!
$   if max_cpu .eq. 0 then max_cpu = 60
$!
$   priv = f$setprv("CMKRNL")
$   set mess/noid/nosev/nofac/notext
$   define/job killed "YES"
$   proc_name = "  " + f$extract(0,13,f$parse(source,,,"NAME")) ! A for Ada
$   set mess/id/sev/fac/text
$!
$   spawn/nowait/nolog/process="'proc_name'" -
        @pcom:run_compiler.bat -
                "'source'"  -
                "'options'" -
            "'debug_option'" -
                "'traces'" -
            "'qualification'"
$!
$! Search the PID of the subprocess
$!
$   set mess/notext/noid/nosev/nofac
$   context = 0
$loop1:
$    pid = f$pid(context)
$    name = f$getjpi( pid , "PRCNAM" )
$    if name .eqs. proc_name then goto loop2
$    if context .ne. 0 then goto loop1
$!
$   write sys$output "Sub-process not found"
$   goto crash_ok
$!
$!  Watch at the log file and at the cpu time
$!
$loop2:
$   cpu = f$getjpi( pid, "CPUTIM" )
$   if $severity .eq. 0 then goto terminated     ! does not exist any more
$   cpu = cpu / 100                              ! put cpu time in second
$   if cpu .gt. max_cpu then goto kill_it
```

```
_DUA1:[PRODUCTIZATION.COMMANDS]COMPILE_IT.PAT;32                    29-0

$   wait 0:0:10                                  ! waits 10 seconds
$   goto loop2
$!
$kill_it:
$   stop/id='pid
$   goto crash_ok
$!
$terminated:                                  ! normal terminaison
$   define/job killed "%D"
$!
$crash_ok:
$   set on
$   set mess/text/fu/fac/sev
$   exit
```

```
$ set noon
$!
$ define sys$output temp:compile.log
$!
$ source        = "'p1'"
$ options       = "'p2'"
$ debug_option  = "'p3'"
$ traces        = "'p4'"
$ qualification = "'p5'"
$!
$ if debug_option then goto debug
$!
$! not debug
$  @dcom:setup
$  ada68k set nolib
$  goto compile
$!
$ debug:
$  @dcom:setup
$  ada68k set nolib
$!
$ compile:
$!
$ priv= f$setprv("CMKRNL")                          ! mandatory
$ set mess/notext/noid/nosev/nofa
$ set working_set /quota=2000/extent=2000           ! set to max authorized
$ define/job err_level "6"                          ! in case of command abort
$ set mess/text/id/sev/fac
$!
$ define/user sys$error sys$output
$!
$ source_name = f$parse(source,,,"name")
$ nom_listing = "resu:" + source_name + ".lis"
$ option_lis  = "/lis='nom_listing'"
$ options = p2 + option_lis
$!
$ if .not. qualification then goto after
$    nom_diag    = "resu:" + source_name + ".dia"
$    option_diag = "/diag='nom_diag'"
$    options     = option + option_diag
$!
$ after:
$!
$ if debug_option .and. traces .nes. "" -
             then options = options + "/trace=(" + traces + ")"
$
$  ada68k compile 'source' 'options'
$!
$ err = $severity
$ set mess/notext/noid/nosev/nofa
$ define/job err_level "'err'"
$ set mess/text/id/sev/fac
$!
$ deassign sys$output
$ delete temp:compile.log;
```

```
10 12:16 1985  adarun Page 1


:1
hell -- not C shell -- script to link, execute an object file and to create
 report file
1 must be a file name without suffix.

_dir=/usr/qualif/debug

test_dir

he object file is linked.
 ld -o exe/$1.e obj/$1.o rts/libada_d.a -lc 2>ldtraces/$1.l \
3 rm obj/$1.o

he exe file is executed, the results of its execution
s in the same named file under the directory
test_dir/results.
1 stands for results, .t stands for traces
he execution is performed in the directory tmp if temp files are created.

test_dir/tmp
 ../exe/$1.e >../results/$1.l 2>../traces/$1.t

 is determined whether the test passed or failed by
arching the string FAILED or PASSED in the result file.

test_dir
grep -c PASSED results/$1.l >/dev/null

cho "$1 passed"
 The execution listing is no more necessary, remove it.
 NO! now keep it for validation.
 rm results/$1.l
 fgrep -c FAILED results/$1.l >/dev/null

cho "$1 failed"

cho "$1 strange"

 The initial object file and the exe file are removed.
xe/$1.e
```

```
 5 15:27 1985  getall Page 1


ell script to run tests as soon as they are transmitted by kermit.
e argument $1 is the name of the train; usually a date such as sep_27.
e files $1.kr and $1.ks contain the outputs of kermit (kept for safety);
e file $1.res contains one status line per test executed.

nore hangup
 "" 1

1?'Give an identifier as parameter (train name)'>

_dir=/usr/qualif/debug

test_dir/stats

d $test_dir/obj ; time /usr/tools/kermit rwilb /dev/tty11 9600 ) \
tee $1.kr \
$test_dir/com/runthem 2>$1.err \
tee $1.res \
$test_dir/com/send_it >$1.ks ) &

 "Transfer begins, cat $1.res and $1.err to see how it goes..."
```

```
30 15:49 1985  runthem Page 1


 Reads messages emitted by kermit on standard input (through the pipeline).
d executes the corresponding files transfered from the VAX.
 The results are written by adarun on standard output.

_dir=/usr/qualif/debug

test_dir/com

ad the kermit version:
 >/dev/null
ad the first file name:
 kermit mss vaxname as name

 loop, while receiving a file name, execute the previous file,
lly transmitted from the host machine:
e [ $mss = Receiving ]

 exit if $name is null:
f [ ! $name ]
hen echo 'Unexpected kermit message'
    exit 1
i
 wait for another file name (or for the message "done" ?) :
f read kermit mss vaxname as newname
hen
  # execute the previous file:
  name=`expr $name : '\([a-z0-9_]*\)'`  # remove the suffix
  adarun $name
  name=$newname
i


 Treat the last transmitted one:
 exit if $name is null:
f [ ! $name ]
hen echo 'Unexpected kermit message'
    exit 1
i
darun $name

 $mss != 'done.' ]

cho 'Unexpected end of kermit transmission'
xit 1
```

APPENDIX D

COMPLETE LIST OF TESTS AND RESULTS

This Appendix presents a complete list of the ACVC test files used in the validation attempt, presented in order by ACVC Implementers' Guide section and objective. Each test name indicates the class of the test and which test objective in the ACVC Implementers' Guide applies to the test.

Each test has a name that identifies the section of the Ada Standard *addressed by the test objective.* The name of a test is interpreted according to the table below, where the first column indicates the character position in the name and the second column, the meaning of that position:

POS	MEANING
1	Test class: A, B, C, D, E, L.
2	Implementers' Guide chapter number (in hexadecimal).
3	Implementers' Guide section number within a chapter (in Hexadecimal)
4	Implementers' Guide subsection number (in hexadecimal)
5-6	Implementers' Guide Test Objective number (in decimal)
7	Test sequence letter
8	[Optional] Compilation sequence digit or letter
9	[Optional] Main program designator in the case of a test having multiple compilation units.

Characters 8 and 9 are only present for tests that consist of several separately compiled units. A series of separately compiled units is counted as one test for reporting purposes. The eighth character indicates

the order in which the units are to be compiled, with unit 0 being compiled first. The ninth character is only present for a file containing a main program for a test comprising multiple files and is always M.

The suffix -AB means the test was written prior to release of the ANSI Standard and is also valid for the version of Ada published in July 1980.

The suffix -B means the test was written specifically for the ANSI Standard. Tests without a suffix have not yet had their names revised to -AB.

A file name ending with the extension .TST indicates that the test depends on one or more of the implementation-dependent parameters listed in Appendix B. A file name ending with .DEP indicates that the test is not necessarily applicable to all implementations because it depends upon the support of language features that a compiler may legally not implement.

The result for each file in ACVC Version 1.6 is given in the following pages, where:

P indicates Passed.

F indicates Failed.

N/A indicates Not Applicable to this implementation.

W indicates Withdrawn due to test errors.

C indicates Compiled without error.

A indicates Anomalous.

A test may comprise several separate compilation units contained in two or more files; the names of such files are indented under the name of the test. The letter 'M' indicates which of these files contains the main procedure.

Support Units

CHECK_FILE-B.ADA	P
REPORT_SPEC-AB.ADA	P
REPORT_BODY-B.ADA	P
VAR_STRINGS_SPEC.ADA	P
VAR_STRINGS_BODY.ADA	P
CZ1101A-AB.ADA	P
CZ1102A-AB.ADA	P
CZ1103A-B.ADA	P
CZ1201A-AB.ADA	P
CZ1201B-AB.ADA	P
CZ1201C-AB.ADA	P
CZ1201D-AB.ADA	P

Chapter 2

A21001A.ADA	P	B23002A.ADA	P	C24113C-B.DEP	N/A
A22002A.ADA	P	B23003D-AB.TST	P	C24113D-B.DEP	N/A
A26004A.TST	P	B23003E-AB.TST	P	C24113E-B.DEP	N/A
A29002A-B.ADA	P	B23003F-AB.TST	P	C24113F-B.DEP	N/A
A29002B-B.ADA	P	B23004A.ADA	P	C24113G-B.DEP	N/A
A29002C-B.ADA	P	B23004B.ADA	P	C24113H-B.DEP	N/A
A29002D-B.ADA	P	B24001A.ADA	P	C24113I-B.DEP	N/A
A29002E-B.ADA	P	B24001B.ADA	P	C24113J-B.DEP	N/A
A29002F-B.ADA	P	B24001C.ADA	P	C24113K-B.DEP	N/A
A29002G-B.ADA	P	B24005A.ADA	P	C24113L-B.DEP	N/A
A29002H-B.ADA	P	B24005B.ADA	P	C24113M-B.DEP	N/A
A29002I-B.ADA	P	B24104A.ADA	P	C24113N-B.DEP	N/A
A29002J-B.ADA	P	B24104B.ADA	P	C24113O-B.DEP	N/A
B22001A-AB.TST	P	B24104C.ADA	P	C24113P-B.DEP	N/A
B22001B-AB.TST	P	B26002A.ADA	P	C24113Q-B.DEP	N/A
B22001C-AB.TST	P	B26005A.ADA	P	C24113R-B.DEP	N/A
B22001D-AB.TST	P	B29001A-B.ADA	P	C24113S-B.DEP	N/A
B22001E-AB.TST	P	C23001A.ADA	P	C24113T-B.DEP	N/A
B22001F-AB.TST	P	C23003A.TST	P	C24113U-B.DEP	N/A
B22001G-AB.TST	P	C24002A.ADA	P	C24113V-B.DEP	N/A
B22001H-AB.ADA	P	C24002B.ADA	P	C24113W-B.DEP	N/A
B22001I-AB.TST	P	C24002C.ADA	P	C24113X-B.DEP	N/A
B22001J-AB.TST	P	C24003A.TST	P	C24113Y-B.DEP	N/A
B22001K-AB.TST	P	C24003B.TST	P	C26002B.ADA	P
B22001L-AB.TST	P	C24003C.TST	P	C26006A-AB.ADA	P
B22001M-AB.TST	P	C24102A.ADA	P	C26008A-AB.ADA	P
B22001N-AB.TST	P	C24102B.ADA	P	C27001A-AB.ADA	P
B22003A.ADA	P	C24102C.ADA	P	C27002A-B.ADA	P
B22004A.ADA	P	C24103A.ADA	P	D29002K-B.ADA	P
B22004B.ADA	P	C24113A-B.DEP	P	E24101A-B.TST	P
B22004C.ADA	P	C24113B-B.DEP	P		

Chapter 3

A32203B-B.ADA	P	B37202A.ADA	P	C35504A-AB.ADA	P
A32203C-B.ADA	P	B37202B.ADA	P	C35504B-B.ADA	P
A32203D-B.ADA	P	B37203A.ADA	P	C35505A.ADA	P
A34008B-B.ADA	P	B37204A-AB.ADA	P	C35505B.ADA	P
A38106D-B.ADA	P	B37205A-AB.ADA	P	C35508A-AB.ADA	P
A38106E-B.ADA	P	B37301A.ADA	P	C35508B-B.ADA	P
B32103A-AB.ADA	P	B37301B.ADA	P	C35702A-AB.DEP	N/A
B32106A-B.ADA	P	B37302A-AB.ADA	P	C35702B-AB.DEP	N/A
B32201A-B.ADA	P	B37303A.ADA	P	C35703A.ADA	P
B32202A-B.ADA	P	B37307B-AB.ADA	P	C35704A-AB.ADA	P
B32202B-B.ADA	P	B37309B-AB.ADA	P	C35704B-AB.ADA	P
B32202C-B.ADA	P	B37310B-B.ADA	P	C35704C-AB.ADA	P
B33001A.ADA	P	B37311A-AB.ADA	P	C35704D-AB.ADA	P
B33002A.ADA	P	B38001A.ADA	P	C35705A-B.DEP	P
B33003A.ADA	P	B38003A-AB.ADA	P	C35705B-B.DEP	P
B33003B-AB.ADA	P	B38008A-B.ADA	P	C35705C-B.DEP	N/A
B33003C-AB.ADA	P	B38008B-AB.ADA	P	C35705D-B.DEP	N/A
B33004A.ADA	P	B38101A-B.ADA	P	C35705E-B.DEP	N/A
B33006A-B.ADA	P	B38101B-AB.ADA	P	C35705F-B.DEP	N/A
B34001S-AB.ADA	P	B38103A-B.ADA	P	C35705G-B.DEP	N/A
B34008A-B.ADA	P	B38103B-B.ADA	P	C35705H-B.DEP	N/A
B35101A.ADA	P	B38103C-B.ADA	P	C35705I-B.DEP	N/A
B35301A.ADA	P	B38103C0	C	C35705J-B.DEP	N/A
B35501A.ADA	P	B38103C1	C	C35705K-B.DEP	N/A
B35506A.ADA	P	B38103C2	C	C35705L-B.DEP	N/A
B35506B.ADA	P	B38103C3M	C	C35705M-B.DEP	N/A
B35701A.TST	P	B38105A-AB.ADA	P	C35705N-B.DEP	N/A
B35709A.ADA	P	B38105B-AB.ADA	W	C35705O-B.DEP	N/A
B35A03A-B.ADA	P	B38106A-B.ADA	P	C35705P-B.DEP	N/A
B36101A-AB.ADA	P	B38106B-B.ADA	P	C35705Q-B.DEP	N/A
B36102A.ADA	P	C32107B-B.ADA	P	C35705R-B.DEP	N/A
B36103A.ADA	P	C32203A-B.ADA	P	C35705S-B.DEP	N/A
B36105A-B.ADA	P	C34001A-B.ADA	P	C35705T-B.DEP	N/A
B36171A-B.ADA	P	C34001B-B.ADA	P	C35705U-B.DEP	N/A
B36171B-B.ADA	P	C34001C-B.ADA	P	C35705V-B.DEP	N/A
B36171C-AB.ADA	P	C34001D-B.DEP	P	C35705W-B.DEP	N/A
B36171D-AB.ADA	P	C34001E-B.DEP	P	C35705X-B.DEP	N/A
B36171E-AB.ADA	P	C34001F-B.DEP	N/A	C35705Y-B.DEP	N/A
B36171F-AB.ADA	P	C34001G-B.DEP	N/A	C35706A-B.DEP	P
B36171G-AB.ADA	P	C34001H-B.ADA	P	C35706B-B.DEP	P
B36171H-AB.ADA	P	C34001I-B.ADA	P	C35706C-B.DEP	N/A
B36171I-AB.ADA	P	C34001K-B.ADA	P	C35706D-B.DEP	N/A
B36201A-B.ADA	P	C34001L-B.ADA	P	C35706E-B.DEP	N/A
B37003A-AB.ADA	P	C34001M-B.ADA	P	C35706F-B.DEP	N/A
B37004A-B.ADA	P	C34001N-B.ADA	P	C35706G-B.DEP	N/A
B37004B-B.ADA	P	C34001O-B.ADA	P	C35706H-B.DEP	N/A
B37004C-B.ADA	P	C34001P-B.ADA	P	C35706I-B.DEP	N/A
B37004D-B.ADA	P	C34001Q-B.ADA	P	C35706J-B.DEP	N/A
B37004E-B.ADA	P	C34001R-B.ADA	P	C35706K-B.DEP	N/A
B37004F-B.ADA	P	C34001T-B.ADA	P	C35706L-B.DEP	N/A
B37004G-B.ADA	P	C34002A-B.ADA	P	C35706M-B.DEP	N/A
B37101A.ADA	P	C34002B-B.ADA	P	C35706N-B.DEP	N/A
B37201A.ADA	P	C35104A.ADA	P	C35706O-B.DEP	N/A

C35706P-B.DEP	N/A	C35708K-B.DEP	N/A	C36205A.ADA	P
C35706Q-B.DEP	N/A	C35708L-B.DEP	N/A	C36205B.ADA	P
C35706R-B.DEP	N/A	C35708M-B.DEP	N/A	C36205C.ADA	P
C35706S-B.DEP	N/A	C35708N-B.DEP	N/A	C36205D.ADA	P
C35706T-B.DEP	N/A	C35708O-B.DEP	N/A	C36205E.ADA	P
C35706U-B.DEP	N/A	C35708P-B.DEP	N/A	C36205F.ADA	P
C35706V-B.DEP	N/A	C35708Q-B.DEP	N/A	C36205G.ADA	P
C35706W-B.DEP	N/A	C35708R-B.DEP	N/A	C36205H.ADA	P
C35706X-B.DEP	N/A	C35708S-B.DEP	N/A	C36205I.ADA	P
C35706Y-B.DEP	N/A	C35708T-B.DEP	N/A	C36205J.ADA	P
C35707A-B.DEP	P	C35708U-B.DEP	N/A	C36205K.ADA	P
C35707B-B.DEP	P	C35708V-B.DEP	N/A	C36301A-B.ADA	P
C35707C-B.DEP	N/A	C35708W-B.DEP	N/A	C36301B-AB.ADA	P
C35707D-B.DEP	N/A	C35708X-B.DEP	N/A	C36302A.ADA	P
C35707E-B.DEP	N/A	C35708Y-B.DEP	N/A	C36303A.ADA	P
C35707F-B.DEP	N/A	C35711A-B.ADA	P	C36304A-B.ADA	P
C35707G-B.DEP	N/A	C35802A-B.DEP	P	C36305A-AB.ADA	P
C35707H-B.DEP	N/A	C35802B-B.DEP	P	C37005A.ADA	P
C35707I-B.DEP	N/A	C35802C-B.DEP	N/A	C37007A-AB.ADA	P
C35707J-B.DEP	N/A	C35802D-B.DEP	N/A	C37008A-B.ADA	P
C35707K-B.DEP	N/A	C35802E-B.DEP	N/A	C37008B-B.ADA	P
C35707L-B.DEP	N/A	C35802F-B.DEP	N/A	C37011A-B.ADA	P
C35707M-B.DEP	N/A	C35802G-B.DEP	N/A	C37012A-AB.ADA	P
C35707N-B.DEP	N/A	C35802H-B.DEP	N/A	C37013A-AB.ADA	P
C35707O-B.DEP	N/A	C35802I-B.DEP	N/A	C37103A-AB.ADA	P
C35707P-B.DEP	N/A	C35802J-B.DEP	N/A	C37105A.ADA	P
C35707Q-B.DEP	N/A	C35802K-B.DEP	N/A	C37208A-B.ADA	P
C35707R-B.DEP	N/A	C35802L-B.DEP	N/A	C37208B-AB.ADA	P
C35707S-B.DEP	N/A	C35802M-B.DEP	N/A	C37209A.ADA	P
C35707T-B.DEP	N/A	C35802N-B.DEP	N/A	C37304A-AB.ADA	P
C35707U-B.DEP	N/A	C35802O-B.DEP	N/A	C37305A.ADA	P
C35707V-B.DEP	N/A	C35802P-B.DEP	N/A	C37306A.ADA	P
C35707W-B.DEP	N/A	C35802Q-B.DEP	N/A	C37307A-AB.ADA	P
C35707X-B.DEP	N/A	C35802R-B.DEP	N/A	C37309A-AB.ADA	P
C35707Y-B.DEP	N/A	C35802S-B.DEP	N/A	C37310A-AB.ADA	P
C35708A-B.DEP	P	C35802T-B.DEP	N/A	C38004A.ADA	P
C35708B-B.DEP	P	C35802U-B.DEP	N/A	C38005A-B.ADA	P
C35708C-B.DEP	N/A	C35802V-B.DEP	N/A	C38006A-B.ADA	P
C35708D-B.DEP	N/A	C35802W-B.DEP	N/A	C38007A-B.ADA	P
C35708E-B.DEP	N/A	C35802X-B.DEP	N/A	C38102A-AB.ADA	P
C35708F-B.DEP	N/A	C35802Y-B.DEP	N/A	C38102B-B.ADA	P
C35708G-B.DEP	N/A	C35904A-B.ADA	P	C38102C-B.ADA	P
C35708H-B.DEP	N/A	C36172A-B.ADA	P	E36202A-B.ADA	P
C35708I-B.DEP	N/A	C36174A-B.ADA	P	E36202B-B.ADA	P
C35708J-B.DEP	N/A	C36204A-B.ADA	P	E38104A-B.ADA	P

Chapter 4

B41101A-B.ADA	P	B45208G-AB.ADA	P	C41303N-B.ADA	P
B41101C-AB.ADA	P	B45208H-B.ADA	P	C41303O-B.ADA	P
B41102A-AB.ADA	P	B45208I-B.ADA	P	C41303Q-B.ADA	P
B41102B-B.ADA	P	B45208M-AB.ADA	P	C41303R-B.ADA	P
B41102C-B.ADA	P	B45208N-AB.ADA	P	C41303S-B.ADA	P
B41201A-B.ADA	P	B45208S-AB.ADA	P	C41303U-B.ADA	P
B41201C.ADA	P	B45208T-AB.ADA	P	C41303V-B.ADA	P
B41202A-B.ADA	P	B45261A-AB.ADA	P	C41303W-B.ADA	P
B41202B-AB.ADA	P	B45261B-AB.ADA	P	C41304A-B.ADA	P
B41202C-B.ADA	P	B45261C-AB.ADA	P	C41306A-B.ADA	P
B41202D-B.ADA	P	B45261D-AB.ADA	P	C41306B-B.ADA	P
B41302A-AB.ADA	P	B45402A.ADA	P	C41306C-B.ADA	P
B41302B-AB.ADA	P	B45522A.ADA	P	C42005A-B.ADA	P
B42004A-B.ADA	P	B45533A-AB.ADA	P	C42006A-B.ADA	P
B43101A-B.ADA	P	B48001A-B.ADA	P	C43103A-B.ADA	P
B43201A-B.ADA	P	B48001B-B.ADA	P	C43103B-B.ADA	P
B43201B-B.ADA	P	B48002A-B.ADA	P	C43107A-B.ADA	P
B43201C-B.ADA	P	B48002B-B.ADA	P	C43205A-B.ADA	P
B43201D-B.ADA	P	B48002C-B.ADA	P	C43205B-B.ADA	P
B43202A-B.ADA	P	B48002D-B.ADA	P	C43205C-B.ADA	P
B43202B-B.ADA	P	B48002E-B.ADA	P	C43205D-B.ADA	P
B43202C-B.ADA	P	B48002F-B.ADA	P	C43205E-B.ADA	P
B43203A-B.ADA	P	B48002G-B.ADA	P	C43205F-B.ADA	P
B43203B-B.ADA	P	B48003A-B.ADA	P	C43205G-B.ADA	P
B44001A-B.ADA	P	B48003B-B.ADA	P	C43205H-B.ADA	P
B44002A-B.ADA	P	B48003C-B.ADA	P	C43205I-B.ADA	P
B44002B-B.ADA	P	B48003D-B.ADA	P	C43205J-B.ADA	P
B44002C.ADA	P	B48003E-B.ADA	P	C43205K-B.ADA	P
B45102A-AB.ADA	P	B4A006A-B.ADA	P	C43206A-B.ADA	P
B45203A.ADA	P	B4A016A.ADA	P	C43207A-B.ADA	P
B45203B-AB.ADA	P	C41101D-B.ADA	P	C43207B-B.ADA	P
B45205A-AB.ADA	P	C41103A-B.ADA	P	C43207C-B.ADA	P
B45206A-AB.ADA	P	C41103B-B.ADA	P	C43207D-B.ADA	P
B45206B-B.ADA	P	C41105A-B.ADA	P	C43208A-B.ADA	P
B45207A-AB.ADA	P	C41106A-B.ADA	P	C43208B-B.ADA	P
B45207B-B.ADA	P	C41107A-AB.ADA	P	C43210A-B.ADA	P
B45207C-B.ADA	P	C41201D-B.ADA	P	C43211A-B.ADA	P
B45207D-B.ADA	P	C41203A-B.ADA	P	C43212A-B.ADA	P
B45207G-B.ADA	P	C41203B-B.ADA	P	C43212C-B.ADA	P
B45207H-B.ADA	P	C41204A.ADA	P	C43213A-B.ADA	P
B45207I-B.ADA	P	C41205A-B.ADA	P	C43214A-B.ADA	P
B45207J-B.ADA	P	C41206A.ADA	P	C43214B-B.ADA	P
B45207M-AB.ADA	P	C41301A-B.ADA	P	C43214C-B.ADA	P
B45207N-AB.ADA	P	C41303A-B.ADA	P	C43214D-B.ADA	P
B45207O-AB.ADA	P	C41303B-B.ADA	P	C43214E-B.ADA	P
B45207P-B.ADA	P	C41303C-B.ADA	P	C43214F-B.ADA	P
B45207S-AB.ADA	P	C41303E-B.ADA	P	C43215A-B.ADA	P
B45207T-AB.ADA	P	C41303F-B.ADA	P	C43215B-B.ADA	P
B45207U-AB.ADA	P	C41303G-B.ADA	P	C45101A.ADA	P
B45207V-B.ADA	P	C41303I-B.ADA	P	C45101B.ADA	P
B45208A-AB.ADA	P	C41303J-B.ADA	P	C45101C.ADA	P
B45208B-B.ADA	P	C41303K-B.ADA	P	C45101E.ADA	P
B45208C-B.ADA	P	C41303M-B.ADA	P	C45101G-AB.ADA	P

C45101H-AB.ADA	P	C45321K-B.DEP	N/A	C45424J-B.DEP	N/A
C45101I.ADA	P	C45321L-B.DEP	N/A	C45424K-B.DEP	N/A
C45103A-AB.ADA	P	C45321M-B.DEP	N/A	C45424L-B.DEP	N/A
C45103B-AB.ADA	P	C45321N-B.DEP	N/A	C45424M-B.DEP	N/A
C45103C-AB.ADA	P	C45321O-B.DEP	N/A	C45424N-B.DEP	N/A
C45104A.ADA	P	C45321P-B.DEP	N/A	C45424O-B.DEP	N/A
C45105A-AB.ADA	P	C45321Q-B.DEP	N/A	C45424P-B.DEP	N/A
C45105B-B.ADA	P	C45321R-B.DEP	N/A	C45424Q-B.DEP	N/A
C45106A.ADA	P	C45321S-B.DEP	N/A	C45424R-B.DEP	N/A
C45201A.ADA	P	C45321T-B.DEP	N/A	C45424S-B.DEP	N/A
C45201B.ADA	P	C45321U-B.DEP	N/A	C45424T-B.DEP	N/A
C45202A-AB.ADA	P	C45321V-B.DEP	N/A	C45424U-B.DEP	N/A
C45210A.ADA	P	C45321W-B.DEP	N/A	C45424V-B.DEP	N/A
C45220A.ADA	P	C45321X-B.DEP	N/A	C45424W-B.DEP	N/A
C45220B.ADA	P	C45321Y-B.DEP	N/A	C45424X-B.DEP	N/A
C45220C.ADA	P	C45342A-AB.ADA	P	C45424Y-B.DEP	N/A
C45220D.ADA	P	C45343A-AB.ADA	P	C45505A-B.ADA	P
C45220E-B.ADA	P	C45345A-AB.ADA	P	C45521A-B.DEP	W
C45241A-B.DEP	P	C45345B-AB.ADA	P	C45521B-B.DEP	W
C45241B-B.DEP	P	C45345C-AB.ADA	P	C45521C-B.DEP	W
C45241C-B.DEP	N/A	C45345D-AB.ADA	P	C45521D-B.DEP	W
C45241D-B.DEP	N/A	C45401A.ADA	P	C45521E-B.DEP	W
C45241E-B.DEP	N/A	C45401B-AB.ADA	P	C45521F-B.DEP	W
C45241F-B.DEP	N/A	C45413A-B.ADA	P	C45521G-B.DEP	W
C45241G-B.DEP	N/A	C45421A-B.DEP	P	C45521H-B.DEP	W
C45241H-B.DEP	N/A	C45421B-B.DEP	P	C45521I-B.DEP	W
C45241I-B.DEP	N/A	C45421C-B.DEP	N/A	C45521J-B.DEP	W
C45241J-B.DEP	N/A	C45421D-B.DEP	N/A	C45521K-B.DEP	W
C45241K-B.DEP	N/A	C45421E-B.DEP	N/A	C45521L-B.DEP	W
C45241L-B.DEP	N/A	C45421F-B.DEP	N/A	C45521M-B.DEP	W
C45241M-B.DEP	N/A	C45421G-B.DEP	N/A	C45521N-B.DEP	W
C45241N-B.DEP	N/A	C45421H-B.DEP	N/A	C45521O-B.DEP	W
C45241O-B.DEP	N/A	C45421I-B.DEP	N/A	C45521P-B.DEP	W
C45241P-B.DEP	N/A	C45421J-B.DEP	N/A	C45521Q-B.DEP	W
C45241Q-B.DEP	N/A	C45421K-B.DEP	N/A	C45521R-B.DEP	W
C45241R-B.DEP	N/A	C45421L-B.DEP	N/A	C45521S-B.DEP	W
C45241S-B.DEP	N/A	C45421M-B.DEP	N/A	C45521T-B.DEP	W
C45241T-B.DEP	N/A	C45421N-B.DEP	N/A	C45521U-B.DEP	W
C45241U-B.DEP	N/A	C45421O-B.DEP	N/A	C45521V-B.DEP	W
C45241V-B.DEP	N/A	C45421P-B.DEP	N/A	C45521W-B.DEP	W
C45241W-B.DEP	N/A	C45421Q-B.DEP	N/A	C45521X-B.DEP	W
C45241X-B.DEP	N/A	C45421R-B.DEP	N/A	C45521Y-B.DEP	W
C45241Y-B.DEP	N/A	C45421S-B.DEP	N/A	C45526A-B.ADA	P
C45264A-B.ADA	P	C45421T-B.DEP	N/A	C45621A.DEP	P
C45274A-AB.ADA	P	C45421U-B.DEP	N/A	C45621B.DEP	P
C45274B-AB.ADA	P	C45421V-B.DEP	N/A	C45621C.DEP	N/A
C45274C-AB.ADA	P	C45421W-B.DEP	N/A	C45621D.DEP	N/A
C45303A-B.ADA	P	C45421X-B.DEP	N/A	C45621E.DEP	N/A
C45321A-B.DEP	P	C45421Y-B.DEP	N/A	C45621F.DEP	N/A
C45321B-B.DEP	P	C45424A-B.DEP	P	C45621G.DEP	N/A
C45321C-B.DEP	N/A	C45424B-B.DEP	P	C45621H.DEP	N/A
C45321D-B.DEP	N/A	C45424C-B.DEP	N/A	C45621I.DEP	N/A
C45321E-B.DEP	N/A	C45424D-B.DEP	N/A	C45621J.DEP	N/A
C45321F-B.DEP	N/A	C45424E-B.DEP	N/A	C45621K.DEP	N/A
C45321G-B.DEP	N/A	C45424F-B.DEP	N/A	C45621L.DEP	N/A
C45321H-B.DEP	N/A	C45424G-B.DEP	N/A	C45621M.DEP	N/A
C45321I-B.DEP	N/A	C45424H-B.DEP	N/A	C45621N.DEP	N/A
C45321J-B.DEP	N/A	C45424I-B.DEP	N/A	C45621O.DEP	N/A

C45621P.DEP	N/A	C48005B-B.ADA	P	C48009H-B.ADA	P
C45621Q.DEP	N/A	C48005C-B.ADA	W	C48009I-B.ADA	P
C45621R.DEP	N/A	C48006A-B.ADA	P	C48009J-B.ADA	P
C45621S.DEP	N/A	C48007B-B.ADA	W	C48010A-B.ADA	P
C45621T.DEP	N/A	C48007A-B.ADA	P	C48012A-B.ADA	P
C45621U.DEP	N/A	C48007B-B.ADA	P	C4A001A.ADA	P
C45621V.DEP	N/A	C48007C-B.ADA	P	C4A003A.ADA	P
C45621W.DEP	N/A	C48008A-B.ADA	P	C4A011A.ADA	P
C45621X.DEP	N/A	C48008B-B.ADA	P	C4A010A-B.ADA	P
C45621Y.DEP	N/A	C48008C-B.ADA	P	C4A013A.ADA	P
C45621Z.DEP	N/A	C48008D-B.ADA	P	D4A002A-AB.ADA	P
C48004A-B.ADA	P	C48009A-B.ADA	P	D4A002B.ADA	P
C48004B-B.ADA	P	C48009B-B.ADA	P	D4A004A-AB.ADA	P
C48004C-B.ADA	P	C48009C-B.ADA	P	D4A004B.ADA	P
C48004D-B.ADA	P	C48009D-B.ADA	P	E43211B-B.ADA	P
C48004E-B.ADA	P	C48009E-B.ADA	P	E43212B-B.ADA	P
C48004F-B.ADA	P	C48009F-B.ADA	P		
C48005A-B.ADA	P	C48009G-B.ADA	P		

Chapter 5

A54B01A-B.ADA	P	B54A27D-AB.ADA	P	B58002B-AB.ADA	P
A54B02A-B.ADA	P	B54B01B-B.TST	P	B58002C-AB.ADA	P
A55B12A-AB.ADA	P	B54B01C-B.ADA	P	B58003A-B.ADA	P
A55B13A-AB.ADA	P	B54B02B-B.ADA	P	B58003B-AB.ADA	P
A55B14A-AB.ADA	P	B54B02C-B.ADA	P	B59001A-AB.ADA	P
B51001A-AB.ADA	P	B54B02D-B.ADA	P	B59001C-AB.ADA	P
B51003A-AB.ADA	P	B54B04A-AB.ADA	P	B59001D-AB.ADA	P
B51004B-B.ADA	P	B54B04B-AB.ADA	P	B59001E-AB.ADA	P
B51004C-B.ADA	P	B54B05A-AB.ADA	P	B59001F-AB.ADA	P
B52002A-B.ADA	P	B55A01A-AB.ADA	P	B59001G-AB.ADA	P
B52002B-AB.ADA	P	B55A01B-AB.ADA	P	B59001H-AB.ADA	P
B52002C-AB.ADA	P	B55A01C-AB.ADA	P	B59001I-AB.ADA	P
B52002D-AB.ADA	P	B55A01D-AB.ADA	P	C51002A-AB.ADA	P
B52002E-AB.ADA	P	B55A01E-AB.ADA	P	C51004A-B.ADA	P
B52002F-B.ADA	P	B55A01F-AB.ADA	P	C52001A-B.ADA	P
B52002G-AB.ADA	P	B55A01G-AB.ADA	P	C52001B-AB.ADA	P
B52003A-AB.ADA	P	B55A01H-AB.ADA	P	C52001C-AB.ADA	P
B52003B-AB.ADA	P	B55A01I-AB.ADA	P	C52005A-AB.ADA	P
B52003C-AB.ADA	P	B55A01J-AB.ADA	P	C52005B-AB.ADA	P
B52004A-B.ADA	P	B55A01K-AB.ADA	P	C52005C-AB.ADA	P
B52004B-AB.ADA	P	B55A01L-AB.ADA	P	C52005D-AB.ADA	P
B52004C-AB.ADA	P	B55A01M-AB.ADA	P	C52005E-AB.ADA	P
B52004D-AB.DEP	P	B55A01N-AB.ADA	P	C52005F-AB.ADA	P
B52004E-AB.DEP	P	B55A01O-AB.ADA	P	C52007A-B.ADA	P
B52006A-AB.ADA	P	B55A01P-AB.ADA	P	C52008A-AB.ADA	P
B53001A-AB.ADA	P	B55A01Q-AB.ADA	P	C52008B-B.ADA	P
B53001B-AB.ADA	P	B55A01R-AB.ADA	P	C52009A-B.ADA	P
B53002A-AB.ADA	P	B55A01S-AB.ADA	P	C52009B-B.ADA	P
B53002B-AB.ADA	P	B55A01T-AB.ADA	P	C52010A-AB.ADA	P
B53003A-AB.ADA	P	B55A01U-AB.ADA	P	C52011A-B.ADA	P
B53004A-AB.ADA	P	B55A01V-AB.ADA	P	C52011B-AB.ADA	P
B53009A-AB.ADA	P	B55B01A-AB.ADA	P	C52012A-AB.ADA	P
B53009B-AB.ADA	P	B55B01B-AB.ADA	P	C52012B-AB.ADA	P
B53009C-AB.ADA	P	B55B09B-AB.ADA	P	C52013A-B.ADA	P
B54A01A-AB.ADA	P	B55B09C-AB.DEP	P	C52101A-AB.ADA	P
B54A01B-AB.ADA	P	B55B09D-AB.DEP	P	C52102A-AB.ADA	P
B54A01C-AB.ADA	P	B55B12B-B.ADA	P	C52102B-AB.ADA	P
B54A01D-AB.ADA	P	B55B12C-AB.ADA	P	C52102C-AB.ADA	P
B54A01E-AB.ADA	P	B55B14B-B.ADA	P	C52102D-AB.ADA	P
B54A01F-AB.ADA	P	B55B18A-B.ADA	P	C52103A-AB.ADA	P
B54A01G-AB.ADA	P	B56001A-AB.ADA	P	C52103B-AB.ADA	P
B54A01H-AB.ADA	P	B56001C-AB.ADA	P	C52103C-AB.ADA	P
B54A01I-AB.ADA	P	B56001D-AB.ADA	P	C52103F-AB.ADA	P
B54A01J-AB.ADA	P	B56001E-AB.ADA	P	C52103G-AB.ADA	P
B54A01K-AB.ADA	P	B56001F-AB.ADA	P	C52103H-AB.ADA	P
B54A01L-AB.ADA	P	B56001G-AB.ADA	P	C52103K-AB.ADA	P
B54A05A.ADA	P	B56001H-AB.ADA	P	C52103L-AB.ADA	P
B54A05B.ADA	P	B57001A-AB.ADA	P	C52103M-AB.ADA	P
B54A08A-B.ADA	P	B57001B-B.ADA	P	C52103P-AB.ADA	P
B54A20A.ADA	P	B57001C-AB.ADA	P	C52103Q-AB.ADA	P
B54A21A-B.ADA	P	B57001D-AB.ADA	P	C52103R-AB.ADA	P
B54A25A-B.ADA	P	B58001A-AB.ADA	P	C52103S-B.ADA	P
B54A27B-AB.ADA	P	B58002A-B.ADA	P	C52103X-B.ADA	N/A

C52104A-AB.ADA	P	C54A27A-AB.ADA	P	C57004B-AB.ADA	P
C52104B-AB.ADA	P	C54A41A.ADA	P	C57004C-AB.ADA	P
C52104C-AB.ADA	P	C54A42A.ADA	P	C57005A-B.ADA	P
C52104F-AB.ADA	P	C54A42B.ADA	P	C58004A-AB.ADA	P
C52104G-AB.ADA	P	C54A42C.ADA	P	C58004B-AB.ADA	P
C52104H-AB.ADA	P	C54A42D.ADA	P	C58004C-AB.ADA	P
C52104K-AB.ADA	P	C54A42E.ADA	P	C58004D-B.ADA	P
C52104L-AB.ADA	P	C54A42F.ADA	P	C58004F-AB.ADA	P
C52104M-AB.ADA	P	C54A42G.ADA	P	C58004G-AB.ADA	P
C52104P-AB.ADA	P	C55B03A-AB.ADA	P	C58005A-AB.ADA	P
C52104Q-AB.ADA	P	C55B04A-AB.ADA	P	C58005B-AB.ADA	P
C52104R-AB.ADA	P	C55B05A-AB.ADA	P	C58005H-AB.ADA	P
C52104X-B.ADA	N/A	C55B06A-AB.ADA	P	C58006A-AB.ADA	P
C52104Y-B.ADA	N/A	C55B06B-AB.ADA	P	C58006B-AB.ADA	P
C53004B-B.ADA	P	C55B07A-AB.DEP	P	C59001B-AB.ADA	P
C53005A-AB.ADA	P	C55B07B-AB.DEP	P	C59002A-AB.ADA	P
C53005B-AB.ADA	P	C55B08A-B.ADA	P	C59002B-AB.ADA	P
C53006A-AB.ADA	P	C55B09A-AB.ADA	P	C59002C-B.ADA	P
C53006B-AB.ADA	P	C55B15A-B.ADA	P	D55A03A-AB.ADA	P
C53007A-AB.ADA	P	C55B16A-AB.DEP	N/A	D55A03B-AB.ADA	P
C53008A-AB.ADA	P	C55C01A-B.ADA	P	D55A03C-AB.ADA	P
C54A03A.ADA	P	C55C02A-AB.ADA	P	D55A03D-AB.ADA	P
C54A04A-AB.ADA	P	C55C02B-AB.ADA	P	D55A03E-AB.ADA	P
C54A06A-AB.ADA	P	C55C03A-AB.ADA	P	D55A03F-AB.ADA	P
C54A07A-AB.ADA	P	C55C03B-AB.ADA	P	D55A03G-AB.ADA	P
C54A22A-AB.ADA	P	C55D01A-AB.ADA	P	D55A03H-AB.ADA	P
C54A23A-B.ADA	P	C56002A-AB.ADA	P	D56001B-AB.ADA	P
C54A24A-AB.ADA	P	C57002A-AB.ADA	P	E52103Y-B.ADA	P
C54A24B.ADA	P	C57003A-AB.ADA	P		
C54A26A.ADA	P	C57004A-AB.ADA	P		

Chapter 6

A62006D-B.ADA	P	B64002A-B.ADA	P	C64104B-AB.ADA	P
A63202A-AB.ADA	P	B64002C-B.ADA	P	C64104C-AB.ADA	P
B61001A-AB.ADA	P	B64003A-B.ADA	P	C64104D-AB.ADA	P
B61001B-AB.ADA	P	B64004A-B.ADA	P	C64104E-AB.ADA	P
B61001C-AB.ADA	P	B64004B-B.ADA	P	C64104F-AB.ADA	P
B61001D-AB.ADA	P	B64004C-B.ADA	P	C64104G-AB.ADA	P
B61001E-AB.ADA	P	B64004D-B.ADA	P	C64104H-B.ADA	P
B61001F-AB.ADA	P	B64004E-B.ADA	P	C64104I-B.ADA	P
B61001G-AB.ADA	P	B64004F-B.ADA	P	C64104J-B.ADA	P
B61001H-AB.ADA	P	B64006A-B.ADA	P	C64104K-AB.ADA	P
B61001I-AB.ADA	P	B64101A-B.ADA	P	C64104L-AB.ADA	P
B61001J-AB.ADA	P	B64201A-B.ADA	P	C64104M-AB.ADA	P
B61001K-AB.ADA	P	B65001A-B.ADA	P	C64104N-B.ADA	P
B61001L-AB.ADA	P	B65002A-AB.ADA	P	C64104O-B.ADA	P
B61001M-AB.ADA	P	B65002B-AB.ADA	P	C64105A-AB.ADA	P
B61001N-AB.ADA	P	B66001A-B.ADA	W	C64105B-AB.ADA	P
B61001O-AB.ADA	P	B66001B-B.ADA	P	C64105C-AB.ADA	P
B61001P-AB.ADA	P	B66001C-B.ADA	P	C64105D-AB.ADA	P
B61001Q-AB.ADA	P	B67001A-B.ADA	W	C64105E-AB.ADA	W
B61001R-AB.ADA	P	B67001B-B.ADA	P	C64105F-AB.ADA	W
B61001S-AB.ADA	P	B67001C-B.ADA	P	C64106A-B.ADA	P
B61001T-AB.ADA	P	B67001D-B.ADA	P	C64106B-B.ADA	P
B61001U-AB.ADA	P	B67001E-B.ADA	P	C64106C-B.ADA	P
B61001V-AB.ADA	P	B67001F-B.ADA	P	C64106D-B.ADA	P
B61001W-AB.ADA	P	B67001G-B.ADA	P	C64107A-B.ADA	P
B61003A-AB.ADA	P	B67004A-B.ADA	W	C64108A-B.ADA	P
B61006A-B.ADA	P	C61003B-AB.ADA	P	C64201B-B.ADA	P
B61011A-B.ADA	P	C61008A-B.ADA	P	C64201C-B.ADA	P
B61012A-B.ADA	P	C61009A-B.ADA	P	C64202A-B.ADA	P
B62001A-AB.ADA	P	C61010A-AB.ADA	P	C65003A-B.ADA	P
B62001B-AB.ADA	P	C62002A-B.ADA	P	C65003B-B.ADA	P
B62001C-AB.ADA	P	C62003A-B.ADA	P	C66002A-B.ADA	P
B62001D-AB.ADA	P	C62003B-B.ADA	P	C66002C-AB.ADA	P
B62006B-B.ADA	P	C62004A-AB.ADA	P	C66002D-AB.ADA	P
B62006C-B.ADA	P	C62006A-B.ADA	P	C66002E-AB.ADA	P
B62006E-B.ADA	P	C63004A-AB.ADA	P	C66002F-AB.ADA	P
B62006F-B.ADA	P	C64002B-B.ADA	P	C66002G-B.ADA	P
B63001A-AB.ADA	P	C64004C-B.ADA	P	C67002A-B.ADA	P
B63001B-AB.ADA	P	C64005A-B.ADA	P	C67002B-B.ADA	P
B63005A-AB.ADA	P	C64005B-B.ADA	P	C67002C-B.ADA	P
B63005B-AB.ADA	P	C64005C-B.ADA	P	C67002D-B.ADA	P
B63005C-AB.ADA	P	C64005D-B.ADA	P	C67002E-B.ADA	P
B63009A-B.ADA	P	C64005D0M	C	C67003A-B.ADA	P
B63009B-B.ADA	P	C64005DA	C	C67003B-B.ADA	P
B63009C-B.ADA	P	C64005DB	C	C67003C-AB.ADA	P
B63009C0	C	C64005DC	C	C67003D-B.ADA	P
B63009C1	C	C64103A-B.ADA	P	C67003E-AB.ADA	P
B63009C2	C	C64103B-B.ADA	P	C67005A-B.ADA	P
B63009C3M	C	C64103C-B.ADA	W	C67005B-B.ADA	P
B63010A-AB.ADA	P	C64103D-B.ADA	W	C67005C-B.ADA	P
B63102A-B.ADA	P	C64103E-B.ADA	P	C67005D-B.ADA	P
B63103A-B.ADA	P	C64103F-B.ADA	P	D64005E-B.ADA	P
B64001A-B.ADA	P	C64104A-AB.ADA	P	D64005E0M	C

D64005EA	C	D64005FF	C	D64005GG	C
D64005EB	C	D64005FG	C	D64005GH	C
D64005EC	C	D64005FH	C	D64005GI	C
D64005ED	C	D64005FI	C	D64005GJ	C
D64005EE	C	D64005FJ	C	D64005GK	C
D64005EF	C	D64005G-B.ADA	P	D64005GL	C
D64005F-B.ADA	P	D64005G0M	C	D64005GM	C
D64005F0M	C	D64005GA	C	D64005GN	C
D64005FA	C	D64005GB	C	D64005GO	C
D64005FB	C	D64005GC	C	D64005GP	C
D64005FC	C	D64005GD	C	D64005GQ	C
D64005FD	C	D64005GE	C		
D64005FE	C	D64005GF	C		

Chapter 7

A71002A-AB.ADA P	B71001Q-AB.ADA P	B74105A-B.ADA P
A71004A-AB.ADA P	B71001R-AB.ADA P	B74105C-B.ADA P
A72001A-AB.ADA P	B71001T-AB.ADA P	B74201A-AB.ADA P
A73001I-AB.ADA P	B71001U-AB.ADA P	B74205A-B.ADA P
A73001J-AB.ADA P	B71001V-AB.ADA P	B74205B-B.ADA P
A74006A-AB.ADA P	B71001W-AB.ADA P	B74207A-B.ADA W
A74105B-B.ADA P	B71002B-AB.ADA P	B74301A-B.ADA P
A74106A-AB.ADA P	B73001A-AB.ADA P	B74304A-B.ADA P
A74106B-AB.ADA P	B73001B-AB.ADA P	B74304B-B.ADA P
A74106C-AB.ADA P	B73001C-B.ADA P	B74304C-B.ADA P
A74205E-B.ADA P	B73001D-B.ADA P	B74401A-B.ADA P
A74205F-B.ADA P	B73001E-AB.ADA P	B74401B-B.ADA P
B71001A-AB.ADA P	B73001F-AB.ADA P	B74409A-B.ADA P
B71001B-AB.ADA P	B73001G-B.ADA P	C72001B-AB.ADA P
B71001C-AB.ADA P	B73001H-B.ADA P	C73002A-B.ADA P
B71001D-AB.ADA P	B73006A-AB.ADA P	C74206A-B.ADA P
B71001E-AB.ADA P	B74001A-AB.ADA P	C74207B-B.ADA P
B71001F-AB.ADA P	B74001B-AB.ADA P	C74209A-AB.ADA P
B71001G-AB.ADA P	B74003A-B.ADA P	C74210A-AB.ADA P
B71001H-AB.ADA P	B74101A-B.ADA P	C74211A-B.ADA P
B71001I-AB.ADA P	B74103A-B.ADA P	C74211B-B.ADA P
B71001J-AB.ADA P	B74103B-B.ADA P	C74302A-B.ADA P
B71001K-AB.ADA P	B74103C-B.ADA P	C74305A-B.ADA P
B71001L-AB.ADA P	B74103D-B.ADA P	C74305B-B.ADA P
B71001M-AB.ADA P	B74103E-B.ADA P	C74402A-B.ADA P
B71001N-AB.ADA P	B74103F-B.ADA W	C74402B-B.ADA P
B71001O-AB.ADA P	B74103G-B.ADA P	C74409B-B.ADA P
B71001P-AB.ADA P	B74104A-B.ADA P	

Chapter 8

A83A02A.ADA	P	B86001BO-B.ADA	P	C87A05A-B.ADA	P
A83A02B.ADA	P	B86001BU-B.ADA	P	C87A05B-B.ADA	P
A83A06A-B.ADA	P	B86001BV-B.ADA	P	C87B02A-B.ADA	P
A83C01C.ADA	P	B86001BW-B.ADA	P	C87B02B-B.ADA	P
A83C01D.ADA	P	B86001BX-B.ADA	P	C87B03A-B.ADA	P
A83C01E.ADA	P	B86001C0M-AB.DEP	P	C87B04A-B.ADA	P
A83C01F.ADA	P	B86001CP-AB.DEP	N/A	C87B04B-B.ADA	P
A83C01G.ADA	P	B86001CQ-AB.DEP	N/A	C87B04C-B.ADA	P
A83C01H.ADA	P	B86001CR-AB.DEP	P	C87B05A-B.ADA	P
A83C01I.ADA	P	B86001CS-AB.DEP	P	C87B06A-B.ADA	P
A83C01J.ADA	P	B86001D0M-AB.TST	P	C87B07A-B.ADA	P
A85007D-B.ADA	P	B86001DT-AB.TST	N/A	C87B07B-B.ADA	P
A85013B-B.ADA	P	B87B23B-B.ADA	P	C87B07C-B.ADA	P
B83A01A-AB.ADA	P	B87B48C-B.ADA	P	C87B07D-B.ADA	P
B83A01B-B.ADA	P	C83B02A.ADA	P	C87B07E-B.ADA	P
B83A01C.ADA	P	C83B02B.ADA	P	C87B08A-B.ADA	P
B83A05A-AB.ADA	P	C83C01B.ADA	P	C87B09A-B.ADA	P
B83A06B-B.ADA	P	C83E02A.ADA	P	C87B09B-B.ADA	P
B83A06H-B.ADA	P	C83E02B.ADA	P	C87B09C-B.ADA	P
B83B01A-AB.ADA	P	C83E03A.ADA	P	C87B10A-B.ADA	P
B83B02C.ADA	P	C83E04A.ADA	P	C87B11A-B.ADA	P
B83C01A-AB.ADA	P	C83F01A.ADA	P	C87B11B-B.ADA	P
B83C02A.ADA	P	C83F01B.ADA	P	C87B13A-B.ADA	P
B83E02C-B.ADA	P	C83F01C.ADA	P	C87B14A-B.ADA	P
B83F02A.ADA	P	C83F01C0	P	C87B14B-B.ADA	P
B83F02B.ADA	P	C83F01C1	P	C87B14C-B.ADA	P
B83F04A-AB.ADA	P	C83F01C2M	P	C87B14D-B.ADA	P
B84001A-AB.ADA	P	C83F01D.ADA	P	C87B15A-B.ADA	P
B84002B-B.ADA	P	C83F01D0M	P	C87B16A-B.ADA	P
B84004A-B.ADA	P	C83F01D1	P	C87B17A-B.ADA	P
B84006A-B.ADA	P	C83F03A.ADA	P	C87B18A-B.ADA	P
B85007B-B.ADA	P	C83F03B.ADA	P	C87B18B-B.ADA	P
B85007C-B.ADA	P	C83F03C.ADA	P	C87B19A-B.ADA	P
B85012A-B.ADA	P	C83F03C0	P	C87B23A-B.ADA	P
B85013C-B.ADA	P	C83F03C1	P	C87B24A-B.ADA	P
B85015A-B.ADA	P	C83F03C2M	P	C87B24B-B.ADA	P
B86001A-AB.ADA	P	C83F03D.ADA	P	C87B26B-B.ADA	P
B86001A0	P	C83F03D0M	P	C87B27A-B.ADA	P
B86001A1M	P	C83F03D1	P	C87B28A-B.ADA	P
B86001B0M	P	C84002A-B.ADA	P	C87B29A-B.ADA	P
B86001BA-B.ADA	P	C85007A-B.ADA	P	C87B30A-B.ADA	P
B86001BB-B.ADA	P	C85007E-B.ADA	P	C87B31A-B.ADA	P
B86001BC-B.ADA	P	C85013A-B.ADA	P	C87B32A-B.ADA	P
B86001BD-B.ADA	P	C86001E-B.ADA	P	C87B33A-B.ADA	P
B86001BE-B.ADA	P	C86001F-B.DEP	N/A	C87B34A-B.ADA	P
B86001BF-B.ADA	P	C86002A.ADA	P	C87B34B-B.ADA	P
B86001BG-B.ADA	P	C86002A0	P	C87B34C-B.ADA	P
B86001BH-B.ADA	P	C86002A1	P	C87B35A-B.ADA	P
B86001BI-B.ADA	P	C86002A2M	P	C87B35B-B.ADA	P
B86001BJ-B.ADA	P	C86002B.ADA	P	C87B35C-B.ADA	P
B86001BK-B.ADA	P	C86002B1	P	C87B37A-B.ADA	P
B86001BL-B.ADA	P	C86002B2M	P	C87B37B-B.ADA	P
B86001BM-B.ADA	P	C86003A-B.ADA	P	C87B37C-B.ADA	P

C87B37D-B.ADA	P	C87B42A-B.ADA	P	C87B48B-B.ADA	P
C87B37E-B.ADA	P	C87B43A-B.ADA	P	C87B54A-B.ADA	P
C87B37F-B.ADA	P	C87B44A-B.ADA	P	C87B57A-B.ADA	P
C87B38A-B.ADA	P	C87B45A-B.ADA	P	C87B62A-B.DEP	N/A
C87B39A-B.ADA	P	C87B45C-B.ADA	P	C87B62B-B.DEP	N/A
C87B40A-B.ADA	P	C87B47A-B.ADA	P	C87B62C-B.DEP	N/A
C87B41A-B.ADA	P	C87B48A-B.ADA	P		

Chapter 9

A91002M-B.ADA	N/A	B950AJA-B.ADA	P	C920BAA-B.ADA	P
A95005A.ADA	P	B950BAA-B.ADA	P	C920BBA-B.ADA	P
A97106A-AB.ADA	P	B950DHA-B.ADA	P	C920BIA-B.ADA	P
B91001A-AB.ADA	P	B96002A-B.ADA	P	C93001A-B.ADA	P
B91001B-AB.ADA	P	B96003A-B.ADA	P	C93002A-B.ADA	P
B91001C-AB.ADA	P	B97101A-AB.ADA	P	C93003A-B.ADA	P
B91001D-AB.ADA	P	B97101B-AB.ADA	P	C93005A-B.ADA	P
B91001E-AB.ADA	P	B97101C-AB.ADA	P	C93005B-B.ADA	W
B91001F-AB.ADA	P	B97101D-AB.ADA	P	C93005C-B.ADA	W
B91001G-B.ADA	N/A	B97101E-AB.ADA	P	C93006A-AB.ADA	P
B91002A-B.ADA	P	B97102A-AB.ADA	P	C93007B-B.ADA	W
B91002B-B.ADA	P	B97102B-AB.ADA	P	C930ABA-B.ADA	P
B91002C-B.ADA	P	B97102C-AB.ADA	P	C930AFA-B.ADA	P
B91002D-B.ADA	P	B97102D-AB.ADA	P	C930AJA-B.ADA	P
B91002E-B.ADA	P	B97102E-AB.ADA	P	C930BAA-B.ADA	P
B91002F-B.ADA	P	B97102F-AB.ADA	P	C94001A-B.ADA	P
B91002G-B.ADA	P	B97102G-AB.ADA	P	C94002A-B.ADA	P
B91002H-B.ADA	P	B97102H-AB.ADA	P	C94002B-B.ADA	P
B91002I-B.ADA	P	B97102I-AB.ADA	P	C94003A-B.ADA	P
B91002J-B.ADA	P	B97103A-AB.ADA	P	C94004A-B.ADA	P
B91002K-B.ADA	P	B97103B-AB.ADA	P	C94004B-B.ADA	P
B91002L-B.ADA	P	B97103D-AB.ADA	P	C94004C-B.ADA	P
B91003A-AB.ADA	P	B97103E-AB.ADA	P	C94005A-B.ADA	P
B91004A-B.ADA	P	B97104A-AB.ADA	P	C94005B-B.ADA	P
B910ABA-B.ADA	P	B97104B-AB.ADA	P	C94006A-B.ADA	P
B910ACA-B.ADA	P	B97104C-AB.ADA	P	C94007A-B.ADA	P
B910AEA-B.ADA	P	B97104D-AB.ADA	P	C94007B-B.ADA	P
B910BCA-B.ADA	P	B97104E-AB.ADA	P	C94020A-B.ADA	P
B920ACA-B.ADA	P	B97104F-AB.ADA	P	C94021A-B.ADA	P
B920BJA-B.ADA	P	B97104G-AB.ADA	P	C940ABA-B.ADA	P
B920BDA-B.ADA	P	B97107A-AB.ADA	P	C940ACA-B.ADA	P
B95001A.ADA	P	B97108A-AB.ADA	P	C940ACB-B.ADA	P
B95001B-AB.ADA	P	B97108B-AB.ADA	P	C940ADA-B.ADA	P
B95002A.ADA	P	B97109A-AB.ADA	P	C940AGA-B.ADA	P
B95004A-AB.ADA	P	B97110A-AB.ADA	P	C940AGB-B.ADA	P
B95004B-AB.ADA	P	B97110B-AB.ADA	P	C940AHA-B.ADA	P
B95006A.ADA	P	B97111A-AB.ADA	P	C940AIA-B.ADA	P
B95006B-AB.ADA	P	B99001A-AB.ADA	P	C940BAA-B.ADA	P
B95006C-AB.ADA	P	B99001B-B.ADA	P	C940BBA-B.ADA	P
B95006D-AB.ADA	P	B99002A-B.ADA	P	C95008A-AB.ADA	P
B95007A-AB.ADA	P	B99002B-B.ADA	P	C95009A-B.ADA	P
B95007B-AB.ADA	P	B99002C-B.ADA	P	C95009B.ADA	P
B95020A-B.ADA	P	B99003A-AB.ADA	P	C95010A.ADA	P
B95020B-B.ADA	P	B9A001A-AB.ADA	P	C95011A.ADA	P
B95020B0	C	B9A001B-AB.ADA	P	C95012A-B.ADA	P
B95020B1	C	C900ACA-B.ADA	P	C950.3A-B.ADA	P
B95020B2M	C	C910AHA-B.ADA	P	C95021A-B.ADA	P
B950ABA-B.ADA	P	C910BDA-B.ADA	P	C95022A-B.ADA	P
B950ABB-B.ADA	P	C910BDB-B.ADA	P	C95022B-B.ADA	P
B950ACA-B.ADA	P	C910BDC-B.ADA	P	C95040D-B.ADA	P
B950ADA-B.ADA	P	C92002A.ADA	P	C950ACB-B.ADA	P
B950AFA-B.ADA	P	C92003A.ADA	P	C950BGA-B.ADA	P
B950AHA-B.ADA	P	C920AJA-B.ADA	P	C950BHA-B.ADA	P

C950BJA-B.ADA	P	C96007A-B.ADA	P	C97303A-AB.ADA	P
C950CAA-B.ADA	P	C96008A-B.ADA	P	C97303B-AB.ADA	P
C950CBA-B.ADA	P	C96008B-B.ADA	P	C97304A-B.ADA	P
C950CHA-B.ADA	P	C97113A-B.ADA	P	C9A003A-B.ADA	P
C950CHC-B.ADA	P	C97114A-B.ADA	P	C9A004A-B.ADA	P
C950DEA-B.ADA	P	C97115A-B.ADA	P	C9A005A-B.ADA	P
C950DEB-B.ADA	P	C97201A-AB.ADA	P	C9A006A-B.ADA	P
C950DGA-B.ADA	P	C97201D-AB.ADA	P	C9A007A-B.ADA	P
C96001A-B.ADA	P	C97201E-AB.ADA	P	C9A009A-B.ADA	P
C96004A-B.ADA	P	C97201G-AB.ADA	P	C9A009B-B.ADA	P
C96005A-B.ADA	P	C97201H-AB.ADA	P	C9A009C-B.ADA	P
C96005B-B.TST	P	C97201X-AB.ADA	P	C9A009D-B.ADA	P
C96005C-B.TST	P	C97202A-AB.ADA	P	C9A009E-B.ADA	P
C96005D-B.ADA	P	C97203A-AB.ADA	P	C9A009F-B.ADA	P
C96005E-B.ADA	P	C97203B-AB.ADA	P	C9A009G-B.ADA	P
C96006A-B.ADA	P	C97204A-B.ADA	P	C9A009H-B.ADA	P

Chapter 10

BA1011B-B.ADA	P	BA1101C0	C	BA3001F0M	C
BA1011B0M	C	BA1101C1	C	BA3001F1	C
BA1011B1	C	BA1101C2M	C	BA3001F2	C
BA1011B2	C	BA1101C3	C	BA3001F3	C
BA1011B3	C	BA1101C4	C	BA3006A-B.ADA	P
BA1011B4	C	BA1101C5	C	BA3006A0	C
BA1011B5	C	BA1101D-AB.ADA	P	BA3006A1	C
BA1011B6	C	BA1101E-B.ADA	P	BA3006A2	C
BA1011B7	C	BA1101F-B.ADA	P	BA3006A3	C
BA1011B8	C	BA1101G-B.ADA	P	BA3006A4	C
BA1011C-B.ADA	P	BA1101H-B.ADA	P	BA3006A5	C
BA1011C0M	C	BA1101H0	C	BA3006A6M	C
BA1011C1	C	BA1101H1M	C	BA3006B-B.ADA	P
BA1011C2	C	BA2001A-AB.ADA	P	BA3006B0	C
BA1011C3	C	BA2001B-AB.ADA	P	BA3006B1	C
BA1011C4	C	BA2001C-AB.ADA	P	BA3006B2	C
BA1011C5	C	BA2001D-AB.ADA	P	BA3006B3	C
BA1011C6	C	BA2001E-AB.ADA	P	BA3006B4M	C
BA1011C7	C	BA2001E0M	C	BA3007A-B.ADA	P
BA1011C8	C	BA2001E1	C	BA3007A0	C
BA1020A-B.ADA	P	BA2001E2	C	BA3007A1	C
BA1020A0M	C	BA2001F-AB.ADA	P	BA3007A2	C
BA1020A1	C	BA2001F0M	C	BA3007A3	C
BA1020A2	C	BA2001F1	C	BA3007A4	C
BA1020A3	C	BA2001F2	C	BA3007A5M	C
BA1020A4	C	BA2001G-AB.ADA	P	BA3007B-B.ADA	P
BA1020A5	C	BA2001G0M	C	BA3007B0	C
BA1020A6	C	BA2001G1	C	BA3007B1	C
BA1020A7	C	BA2003B-AB.ADA	P	BA3007B2	C
BA1020A8	C	BA2003B0M	C	BA3007B3	C
BA1020B-B.ADA	P	BA2003B1	C	BA3007B4	C
BA1020B0	C	BA2013A-B.ADA	P	BA3007B5	C
BA1020B1	C	BA2013B-B.ADA	P	BA3007B6	C
BA1020B2	C	BA3001A-AB.ADA	P	BA3007B7	C
BA1020B3	C	BA3001A0M	C	BA3007B8M	C
BA1020B4	C	BA3001A1	C	BA3008A-B.ADA	P
BA1020B5	C	BA3001A2	C	BA3008A0	C
BA1020B6M	C	BA3001A3	C	BA3008A1	C
BA1020C-B.ADA	P	BA3001B.ADA	P	BA3008A2	C
BA1020C0M	C	BA3001B0M	C	BA3008A3	C
BA1020C1	C	BA3001B1	C	BA3008A4	C
BA1020C2	C	BA3001C-AB.ADA	P	BA3008A5M	C
BA1020C3	C	BA3001C0M	C	BA3008B-B.ADA	P
BA1020C4	C	BA3001C1	C	BA3008B0	C
BA1020C5	C	BA3001D-AB.ADA	P	BA3008B1	C
BA1101A-AB.ADA	P	BA3001D0M	C	LA3008B2	C
BA1101B-B.ADA	P	BA3001D1	C	BA3008B3	C
BA1101B0M	C	BA3001E-AB.ADA	P	BA3008B4	C
BA1101B1	C	BA3001E0M	C	BA3008B5	C
BA1101B2	C	BA3001E1	C	BA3008B6M	C
BA1101B3	C	BA3001E2	C	BA3013A-B.ADA	P
BA1101B4	C	BA3001E3	C	BA3013A0	C
BA1101C-B.ADA	P	BA3001F-AB.ADA	P	BA3013A1	C

BA3013A2	C	CA1013A5	C	CA2008A-B.ADA	P
BA3013A3	C	CA1013A6M	C	CA2008A0M	C
BA3013A4	C	CA1014A-AB.ADA	P	CA2008A1	C
BA3013A5	C	CA1014A0M	C	CA2008A2	C
BA3013A6	C	CA1014A1	C	CA2009A-B.DEP	P
BA3013A7M	C	CA1014A2	C	CA2009B-B.DEP	W
CA1002A-B.ADA	P	CA1014A3	C	CA2009C-B.DEP	N/A
CA1002A0	C	CA1022A-B.ADA	P	CA2009C0M	C
CA1002A1	C	CA1022A0	C	CA2009C1	C
CA1002A2	C	CA1022A1	C	CA2009D-B.DEP	P
CA1002A3M	C	CA1022A2	C	CA2009E-B.DEP	W
CA1002A4	C	CA1022A3	C	CA2009F-B.DEP	W
CA1002A5	C	CA1022A4	C	CA2009F0M	C
CA1002A6	C	CA1022A5	C	CA2009F1	C
CA1002A7	C	CA1022A6M	C	CA3002A-B.ADA	P
CA1002A8	C	CA1102A-B.ADA	P	CA3002A0	C
CA1002A9	C	CA1102A0	C	CA3002A1	C
CA1003A-AB.ADA	P	CA1102A1	C	CA3002A2M	C
CA1003B-AB.ADA	W	CA1102A2M	C	CA3002A3	C
CA1004A-AB.ADA	P	CA1105A-B.ADA	P	CA3006C-B.ADA	P
CA1005A-AB.ADA	P	CA1105A0	C	CA3006C0	C
CA1006A-AB.ADA	P	CA1105A1M	C	CA3006C1	C
CA1007A-AB.ADA	P	CA1105B-B.ADA	P	CA3006C2	C
CA1007A0	C	CA1105B0	C	CA3006C3	C
CA1007A1M	C	CA1105B1	C	CA3006C4	C
CA1008A-AB.ADA	P	CA1105B2	C	CA3006C5M	C
CA1008A0	C	CA1105B3M	C	CA3006D-B.ADA	P
CA1008A1M	C	CA1105B4	C	CA3006D0	C
CA1009A-AB.ADA	P	CA1105B5	C	CA3006D1	C
CA1009A0	C	CA1107A.ADA	P	CA3006D2	C
CA1009A1	C	CA1107A0	C	CA3006D3M	C
CA1009A2	C	CA1107A1M	C	CA3006E-B.ADA	P
CA1009A3	C	CA1107A2	C	CA3006E0	C
CA1009A4M	C	CA1108A-B.ADA	W	CA3006E1	C
CA1011A-B.ADA	W	CA1108B-B.ADA	W	CA3006E2	C
CA1011A0	W	CA2001H-B.ADA	P	CA3006E3	C
CA1011A1	W	CA2001H0	C	CA3006E4	C
CA1011A2	W	CA2001H1	C	CA3006E5	C
CA1011A3	W	CA2001H2	C	CA3006E6M	C
CA1011A4	W	CA2001H3M	C	CA5002A-B.ADA	P
CA1011A5	W	CA2002A-B.ADA	P	CA5002B-B.ADA	P
CA1011A6M	W	CA2002A0M	C	CA5002B0	C
CA1012A-B.DEP	P	CA2002A1	C	CA5002B1	C
CA1012A0	C	CA2002A2	C	CA5002B2	C
CA1012A1	C	CA2003A-AB.ADA	P	CA5002B3	C
CA1012A2	C	CA2003A0M	C	CA5002B4	C
CA1012A3	C	CA2003A1	C	CA5002B5	C
CA1012A4M	C	CA2004A-AB.ADA	P	CA5002B6	C
CA1012B-B.ADA	P	CA2004A0M	C	CA5002B7M	C
CA1012B0	C	CA2004A1	C	CA5003A-B.ADA	P
CA1012B2	C	CA2004A2	C	CA5003A0	C
CA1012B4M	C	CA2004A3	C	CA5003A1	C
CA1013A-B.ADA	P	CA2004A4	C	CA5003A2	C
CA1013A0	C	CA2007A-AB.ADA	P	CA5003A3	C
CA1013A1	C	CA2007A0M	C	CA5003A4	C
CA1013A2	C	CA2007A1	C	CA5003A5	C
CA1013A3	C	CA2007A2	C	CA5003A6M	C
CA1013A4	C	CA2007A3	C	CA5003B-B.ADA	P

CA5003B0	C	LA3004A2	C	LA3004B5	C
CA5003B1	C	LA3004A3	C	LA3004B6M	C
CA5003B2	C	LA3004A4	C	LA5001A-B.ADA	P
CA5003B3	C	LA3004A5	C	LA5001A0	C
CA5003B4	C	LA3004A6M	C	LA5001A1	C
CA5003B5M	C	LA3004B-B.ADA	N/A	LA5001A2	C
CA5004A-B.ADA	P	LA3004B0	C	LA5001A3	C
CA5004B-B.ADA	P	LA3004B1	C	LA5001A4	C
LA3004A-AB.ADA	N/A	LA3004B2	C	LA5001A5	C
LA3004A0	C	LA3004B3	C	LA5001A6	C
LA3004A1	C	LA3004B4	C	LA5001A7M	C

Chapter 11

BB2001A-AB.ADA	P	CB1003A-AB.ADA	P	CB4003A-AB.ADA	P
BB2002A-AB.ADA	P	CB1004A-AB.ADA	P	CB4004A-B.ADA	P
BB2003A-AB.ADA	P	CB2004A-B.ADA	P	CB4005A-AB.ADA	P
BB2003B-AB.ADA	P	CB2005A-B.ADA	P	CB4006A-B.ADA	P
BB2003C-AB.ADA	P	CB2006A-AB.ADA	P	CB4008A-AB.ADA	P
BB3001A-B.ADA	P	CB2007A-AB.ADA	P	CB4009A-AB.ADA	P
BB3002A-AB.ADA	P	CB3003A-B.ADA	P	CB5001A-B.ADA	P
BB3005A-AB.ADA	P	CB3004A-AB.ADA	P	CB5001B-B.ADA	P
CB1001A-B.ADA	P	CB4001A-AB.ADA	P		
CB1002A-B.ADA	P	CB4002A-AB.ADA	P		

Chapter 12

BC1001A-B.ADA	P	BC3002A-AB.ADA	P	BC32ADA-B.ADA	P
BC1002A-B.ADA	N/A	BC3002B-AB.ADA	P	BC3301A-AB.ADA	P
BC1008A-AB.ADA	P	BC3002C-AB.ADA	P	BC3301B-AB.ADA	P
BC1008B-AB.ADA	P	BC3002D-AB.ADA	P	BC3302A-AB.ADA	P
BC1008C-AB.ADA	P	BC3002E-AB.ADA	P	BC3302B-AB.ADA	P
BC1009A-AB.ADA	P	BC3003A-AB.ADA	P	BC3303A-AB.ADA	P
BC1011A-AB.ADA	P	BC3003B-AB.ADA	P	BC3304A-AB.ADA	P
BC1011B-AB.ADA	P	BC3005A-AB.ADA	P	BC33ABA-B.ADA	P
BC1012A-AB.ADA	P	BC3006A-AB.ADA	P	BC33ACA-B.ADA	P
BC1013A-B.ADA	W	BC3009A-B.ADA	P	BC33ADA-B.ADA	P
BC10ABA-B.ADA	P	BC3009B-B.ADA	P	BC33AEA-B.ADA	P
BC10ABB-B.ADA	P	BC3009C-B.ADA	P	BC3401A-AB.ADA	P
BC10ACA-B.ADA	P	BC3011B-B.ADA	P	BC3401B-AB.ADA	P
BC10ADA-B.ADA	P	BC3011C-AB.ADA	P	BC3402A-AB.ADA	P
BC10AEA-B.ADA	P	BC3013A-AB.ADA	P	BC3402B-AB.ADA	P
BC10AEB-B.ADA	P	BC3018A-B.ADA	P	BC3403A-AB.ADA	P
BC10AEC-B.ADA	P	BC30ABA-B.ADA	P	BC3403B-AB.ADA	P
BC10AED-B.ADA	P	BC30ACA-B.ADA	P	BC3403C-AB.ADA	P
BC10AFA-B.ADA	P	BC3101A-B.ADA	P	BC3404A-AB.ADA	P
BC10AGA-B.ADA	P	BC3101B-B.ADA	P	BC3404B-B.ADA	P
BC1101A-AB.ADA	P	BC3102A-B.ADA	P	BC3404C-AB.ADA	P
BC1102A-B.ADA	P	BC3102B-B.ADA	P	BC3404D-AB.ADA	P
BC1103A-B.ADA	P	BC3103A-AB.ADA	P	BC3404E-AB.ADA	P
BC1104A-B.ADA	P	BC3103B-AB.ADA	P	BC3404F-AB.ADA	P
BC1104B-B.ADA	P	BC31ABA-B.ADA	P	BC3405A-AB.ADA	P
BC1106A-AB.ADA	P	BC31ACA-B.ADA	P	BC3405B-B.ADA	W
BC1107A-B.ADA	P	BC31ADA-B.ADA	P	BC3405D-AB.ADA	P
BC11ABA-B.ADA	P	BC3201A-B.ADA	P	BC3405E-AB.ADA	P
BC11ACA-B.ADA	P	BC3201B-AB.ADA	P	BC3405F-AB.ADA	P
BC1201A-AB.ADA	P	BC3201C-B.ADA	P	BC3501A-AB.ADA	P
BC1201B-AB.ADA	P	BC3202A-B.ADA	P	BC3501B-AB.ADA	P
BC1201C-AB.ADA	P	BC3202B-B.ADA	P	BC3501C-AB.ADA	P
BC1201D-AB.ADA	P	BC3202C-B.ADA	P	BC3501D-AB.ADA	P
BC1202A-AB.ADA	P	BC3203B-B.ADA	P	BC3501E-AB.ADA	P
BC1202B-AB.ADA	P	BC3204A-B.ADA	W	BC3501F-AB.ADA	P
BC1202C-AB.ADA	P	BC3204B-B.ADA	W	BC3501G-AB.ADA	P
BC1202D-AB.ADA	P	BC3204C-B.ADA	W	BC3501H-AB.ADA	P
BC1203A-AB.ADA	P	BC3204C0	W	BC3501I-AB.ADA	P
BC1207A-B.ADA	P	BC3204C1	W	BC3501J-AB.ADA	P
BC1226A-B.ADA	P	BC3204C2	W	BC3501K-AB.ADA	P
BC12ABA-B.ADA	P	BC3204D-B.ADA	W	BC3502A-AB.ADA	P
BC12ACA-B.ADA	P	BC3204E-B.ADA	P	BC3502B-AB.ADA	P
BC12ACB-B.ADA	P	BC3205A-B.ADA	W	BC3502C-AB.ADA	P
BC1303A-AB.ADA	P	BC3205B-B.ADA	W	BC3502D-AB.ADA	P
BC1303B-AB.ADA	P	BC3205C-B.ADA	W	BC3502E-AB.ADA	P
BC1303C-AB.ADA	P	BC3205D-B.ADA	W	BC3502F-AB.ADA	P
BC1303D-AB.ADA	P	BC3205D0	W	BC3502G-AB.ADA	P
BC1303E-AB.ADA	P	BC3205D1M	W	BC3502H-AB.ADA	P
BC1306A-B.ADA	P	BC3205D2	W	BC3502I-AB.ADA	P
BC13ABA-B.ADA	P	BC3205E-B.ADA	P	BC3502J-AB.ADA	P
BC2001B-AB.ADA	P	BC3205F-B.ADA	P	BC3502K-AB.ADA	P
BC2001C-AB.ADA	P	BC3220B-B.ADA	W	BC3502L-AB.ADA	P
BC20ABA-B.ADA	P	BC32ABA-B.ADA	P	BC3502M-AB.ADA	P

BC3502N-AB.ADA	P	CC3007A-AB.ADA	P	CC3407D-AB.ADA	P
BC3502O-AB.ADA	P	CC3011A-B.ADA	P	CC3407E-AB.ADA	P
BC3503A-B.ADA	W	CC3011D-B.ADA	P	CC3407F-AB.ADA	P
BC3503B-B.ADA	P	CC3012A-AB.ADA	P	CC3408A-AB.ADA	P
BC3503C-B.ADA	P	CC3120A-AB.ADA	P	CC3408B-AB.ADA	P
BC3503D-B.ADA	P	CC3120B-B.ADA	P	CC3408C-AB.ADA	P
BC3503F-B.ADA	P	CC3125A-B.ADA	P	CC3408D-B.ADA	P
CC1004A-AB.ADA	P	CC3203A-B.ADA	P	CC3504A-B.ADA	P
CC1010A-AB.ADA	P	CC3208A-AB.ADA	P	CC3504B-B.ADA	P
CC1010B-AB.ADA	P	CC3208B-AB.ADA	P	CC3504C-B.ADA	P
CC1204A-B.ADA	P	CC3305A-AB.ADA	P	CC3504D-B.ADA	P
CC1220A-B.ADA	P	CC3305B-AB.ADA	P	CC3504E-B.ADA	P
CC1301A-B.ADA	P	CC3305C-AB.ADA	P	CC3504F-B.ADA	P
CC1302A-AB.ADA	P	CC3305D-AB.ADA	P	CC3504G-B.ADA	P
CC1304A-AB.ADA	P	CC3406A-AB.ADA	P	CC3504H-B.ADA	P
CC1305B-AB.ADA	P	CC3406B-AB.ADA	P	CC3504I-B.ADA	P
CC1307A-AB.ADA	P	CC3406C-AB.ADA	P	CC3504J-B.ADA	P
CC1308A-AB.ADA	P	CC3406D-B.ADA	P	CC3504K-B.ADA	P
CC1310A-AB.ADA	P	CC3407A-AB.ADA	P	CC3601C-AB.ADA	P
CC2002A-AB.ADA	P	CC3407B-AB.ADA	P	CC3602A-AB.ADA	P
CC3004A-B.ADA	P	CC3407C-AB.ADA	P		

Chapter 14

AE2101A-B.ADA	P	CE2111D-B.ADA	P	CE3301A-B.ADA	P
AE2101B-B.ADA	P	CE2201A-B.ADA	P	CE3301B-B.ADA	P
AE2101C-B.DEP	P	CE2201B-B.ADA	P	CE3301C-B.ADA	P
AE2101D-B.ADA	P	CE2201C-B.ADA	P	CE3302A-B.ADA	P
AE3101A-B.ADA	P	CE2201D-B.DEP	P	CE3303A-B.ADA	P
AE3702A-B.ADA	P	CE2201E-B.DEP	P	CE3305A-B.ADA	P
AE3709A-B.ADA	P	CE2201F-B.ADA	P	CE3402A-B.ADA	P
BE2101E-B.ADA	P	CE2202A-B.ADA	P	CE3402B-B.ADA	P
BE2112A-B.ADA	P	CE2204A-B.ADA	P	CE3402C-B.ADA	P
BE2112B-B.ADA	P	CE2204B-B.ADA	P	CE3402D-B.ADA	P
BE2112C-B.ADA	P	CE2210A-B.ADA	P	CE3402E-B.ADA	P
BE2114A-B.ADA	P	CE2401A-B.ADA	P	CE3403A-B.ADA	P
BE2208A-B.ADA	P	CE2401B-B.ADA	P	CE3403B-B.ADA	P
BE3001A-B.ADA	P	CE2401C-B.ADA	P	CE3403C-B.ADA	P
BE3002A-B.ADA	P	CE2401D-B.DEP	P	CE3403D-B.ADA	P
BE3002E-B.ADA	P	CE2401E-B.ADA	P	CE3403E-B.ADA	P
BE3105A-B.ADA	P	CE2401F-B.ADA	P	CE3403F-B.ADA	P
BE3205A-B.ADA	P	CE2402A-B.ADA	P	CE3404A-B.ADA	P
BE3501A-B.ADA	P	CE2404A-B.ADA	P	CE3404B-B.ADA	P
BE3606C-B.ADA	P	CE2405B-B.ADA	P	CE3404C-B.ADA	P
BE3703A-B.ADA	P	CE2406A-B.ADA	P	CE3405A-B.ADA	P
BE3802A-B.ADA	P	CE2407A-B.ADA	P	CE3405B-B.ADA	P
BE3803A-B.ADA	P	CE2408A-B.ADA	P	CE3405C-B.ADA	P
BE3902A-B.ADA	P	CE2409A-B.ADA	P	CE3405D-B.ADA	P
BE3903A-B.ADA	P	CE2410A-B.ADA	P	CE3406A-B.ADA	P
CE2102A-B.ADA	P	CE3002B-B.TST	P	CE3406B-B.ADA	P
CE2102B-B.ADA	P	CE3002C-B.TST	P	CE3406C-B.ADA	P
CE2102C-B.TST	P	CE3002D-B.ADA	P	CE3406D-B.ADA	P
CE2102D-B.ADA	P	CE3002F-B.ADA	P	CE3407A-B.ADA	P
CE2102E-B.ADA	P	CE3102A-B.ADA	P	CE3407B-B.ADA	P
CE2102F-B.ADA	P	CE3102B-B.TST	P	CE3407C-B.ADA	P
CE2102G-B.ADA	P	CE3103A-B.ADA	P	CE3408A-B.ADA	P
CE2103A-B.TST	P	CE3104A-B.ADA	P	CE3408B-B.ADA	P
CE2103B-B.TST	P	CE3107A-B.TST	P	CE3408C-B.ADA	P
CE2104A-B.ADA	P	CE3108A-B.ADA	P	CE3409A-B.ADA	P
CE2104B-B.ADA	P	CE3108B-B.ADA	P	CE3409B-B.ADA	P
CE2105A-B.ADA	P	CE3109A-B.ADA	P	CE3409C-B.ADA	P
CE2106A-B.ADA	P	CE3110A-B.ADA	P	CE3409D-B.ADA	P
CE2107A-B.ADA	P	CE3111A-B.ADA	P	CE3409E-B.ADA	P
CE2107B-B.ADA	P	CE3111B-B.ADA	P	CE3409F-B.ADA	P
CE2107C-B.ADA	P	CE3111C-B.ADA	P	CE3410A-B.ADA	P
CE2107D-B.ADA	P	CE3111D-B.ADA	P	CE3410B-B.ADA	P
CE2107E-B.ADA	W	CE3111E-B.ADA	P	CE3410C-B.ADA	P
CE2108A-B.ADA	P	CE3112A-B.ADA	P	CE3410D-B.ADA	P
CE2108B-B.ADA	P	CE3112B-B.ADA	P	CE3410E-B.ADA	P
CE2108C-B.ADA	P	CE3114A-B.ADA	P	CE3410F-B.ADA	P
CE2108D-B.ADA	P	CE3114B-B.ADA	P	CE3411A-B.ADA	P
CE2109A-B.ADA	P	CE3115A-B.ADA	P	CE3411C-B.ADA	P
CE2110A-B.ADA	P	CE3201A-B.ADA	P	CE3412A-B.ADA	P
CE2110B-B.ADA	P	CE3202A-B.ADA	P	CE3412C-B.ADA	P
CE2111A-B.ADA	P	CE3203A-B.ADA	P	CE3413A-B.ADA	P
CE2111B-B.ADA	P	CE3206A-B.ADA	P	CE3413C-B.ADA	P
CE2111C-B.ADA	P	CE3208A-B.ADA	P	CE3601A-B.ADA	P

CE3602A-B.ADA	P	CE3704N-B.ADA	P	CE3806A-B.ADA	P
CE3602B-B.ADA	P	CE3704O-B.ADA	P	CE3806C-B.ADA	P
CE3602C-B.ADA	P	CE3706C-B.ADA	P	CE3806D-B.ADA	P
CE3602D-B.ADA	P	CE3706D-B.ADA	P	CE3806E-B.ADA	P
CE3603A-B.ADA	W	CE3706F-B.ADA	P	CE3809A-B.ADA	P
CE3604A-B.ADA	W	CE3706G-B.ADA	P	CE3809B-B.ADA	P
CE3605A-B.ADA	P	CE3707A-B.ADA	P	CE3810A-B.ADA	P
CE3605B-B.ADA	P	CE3708A-B.ADA	P	CE3901A-B.ADA	P
CE3605C-B.ADA	P	CE3801A-B.ADA	P	CE3905A-B.ADA	P
CE3605D-B.ADA	P	CE3804A-B.ADA	P	CE3905B-B.ADA	P
CE3605E-B.ADA	P	CE3804B-B.ADA	P	CE3905C-B.ADA	P
CE3606A-B.ADA	P	CE3804C-B.ADA	P	CE3905L-B.ADA	P
CE3606B-B.ADA	P	CE3804D-B.ADA	P	CE3906A-B.ADA	P
CE3701A-B.ADA	P	CE3804E-B.ADA	P	CE3906B-B.ADA	P
CE3704A-B.ADA	P	CE3804F-B.ADA	P	CE3906C-B.ADA	P
CE3704B-B.ADA	P	CE3804G-B.ADA	P	CE3906D-B.ADA	P
CE3704C-B.ADA	P	CE3804I-B.ADA	P	CE3906E-B.ADA	P
CE3704D-B.ADA	P	CE3804K-B.ADA	P	CE3906F-B.ADA	P
CE3704E-B.ADA	P	CE3804M-B.ADA	P	CE3907A-B.ADA	P
CE3704F-B.ADA	P	CE3805A-B.ADA	P	CE3908A-B.ADA	P
CE3704M-B.ADA	W	CE3805B-B.ADA	P	EE3102C-B.ADA	P

END 6-86